Binance Testnet – Websocket API

**General API Information**

* The base endpoint is: **wss://ws-api.testnet.binance.vision/ws-api/v3**
  + If you experience issues with the standard 443 port, alternative port 9443 is also available.
* A single connection to the API is only valid for 24 hours; expect to be disconnected after the 24-hour mark.
* Responses are in JSON by default. To receive responses in SBE, refer to the [SBE FAQ](https://developers.binance.com/docs/binance-spot-api-docs/faqs/sbe_faq) page.
* The WebSocket server will send a ping frame every 20 seconds.
  + If the WebSocket server does not receive a pong frame back from the connection within a minute the connection will be disconnected.
  + When you receive a ping, you must send a pong with a copy of ping's payload as soon as possible.
  + Unsolicited pong frames are allowed, but will not prevent disconnection. **It is recommended that the payload for these pong frames are empty.**
* Lists are returned in **chronological order**, unless noted otherwise.
* All timestamps in the JSON responses are in **milliseconds in UTC by default**. To receive the information in microseconds, please add the parameter timeUnit=MICROSECOND or timeUnit=microsecond in the URL.
* Timestamp parameters (e.g. startTime, endTime, timestamp) can be passed in milliseconds or microseconds.
* All field names and values are **case-sensitive**, unless noted otherwise.
* If there are enums or terms you want clarification on, please see [SPOT Glossary](https://developers.binance.com/docs/binance-spot-api-docs/faqs/spot_glossary) for more information.
* APIs have a timeout of 10 seconds when processing a request. If a response from the Matching Engine takes longer than this, the API responds with "Timeout waiting for response from backend server. Send status unknown; execution status unknown." [(-1007 TIMEOUT)](https://developers.binance.com/docs/binance-spot-api-docs/testnet/errors#-1007-timeout)
  + This does not always mean that the request failed in the Matching Engine.
  + If the status of the request has not appeared in [User Data Stream](https://developers.binance.com/docs/binance-spot-api-docs/testnet/user-data-stream), please perform an API query for its status.

**Request format**

Requests must be sent as JSON in **text frames**, one request per frame.

Example of request:

{  
 "id": "e2a85d9f-07a5-4f94-8d5f-789dc3deb097",  
 "method": "order.place",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "BUY",  
 "type": "LIMIT",  
 "price": "0.1",  
 "quantity": "10",  
 "timeInForce": "GTC",  
 "timestamp": 1655716096498,  
 "apiKey": "T59MTDLWlpRW16JVeZ2Nju5A5C98WkMm8CSzWC4oqynUlTm1zXOxyauT8LmwXEv9",  
 "signature": "5942ad337e6779f2f4c62cd1c26dba71c91514400a24990a3e7f5edec9323f90"  
 }  
}

Request fields:

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| id | INT / STRING / null | YES | Arbitrary ID used to match responses to requests |
| method | STRING | YES | Request method name |
| params | OBJECT | NO | Request parameters. May be omitted if there are no parameters |

* Request id is truly arbitrary. You can use UUIDs, sequential IDs, current timestamp, etc. The server does not interpret id in any way, simply echoing it back in the response.

You can freely reuse IDs within a session. However, be careful to not send more than one request at a time with the same ID, since otherwise it might be impossible to tell the responses apart.

* Request method names may be prefixed with explicit version: e.g., "v3/order.place".
* The order of params is not significant.

**Response format**

Responses are returned as JSON in **text frames**, one response per frame.

Example of successful response:

{  
 "id": "e2a85d9f-07a5-4f94-8d5f-789dc3deb097",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "orderId": 12510053279,  
 "orderListId": -1,  
 "clientOrderId": "a097fe6304b20a7e4fc436",  
 "transactTime": 1655716096505,  
 "price": "0.10000000",  
 "origQty": "10.00000000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "BUY",  
 "workingTime": 1655716096505,  
 "selfTradePreventionMode": "NONE"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 12  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 4043  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 321  
 }  
 ]  
}

Example of failed response:

{  
 "id": "e2a85d9f-07a5-4f94-8d5f-789dc3deb097",  
 "status": 400,  
 "error": {  
 "code": -2010,  
 "msg": "Account has insufficient balance for requested action."  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 13  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 4044  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 322  
 }  
 ]  
}

Response fields:

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| id | INT / STRING / null | YES | Same as in the original request |
| status | INT | YES | Response status. See [Status codes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/response-format#status-codes) |
| result | OBJECT / ARRAY | YES | Response content. Present if request succeeded |
| error | OBJECT | Error description. Present if request failed |
| rateLimits | ARRAY | NO | Rate limiting status. See [Rate limits](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/response-format#rate-limits) |

**Status codes**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/response-format#status-codes)

Status codes in the status field are the same as in HTTP.

Here are some common status codes that you might encounter:

* 200 indicates a successful response.
* 4XX status codes indicate invalid requests; the issue is on your side.
  + 400 – your request failed, see error for the reason.
  + 403 – you have been blocked by the Web Application Firewall.
  + 409 – your request partially failed but also partially succeeded, see error for details.
  + 418 – you have been auto-banned for repeated violation of rate limits.
  + 429 – you have exceeded API request rate limit, please slow down.
* 5XX status codes indicate internal errors; the issue is on Binance's side.
  + **Important:** If a response contains 5xx status code, it **does not** necessarily mean that your request has failed. Execution status is *unknown* and the request might have actually succeeded. Please use query methods to confirm the status. You might also want to establish a new WebSocket connection for that.

See [Error codes for Binance](https://developers.binance.com/docs/binance-spot-api-docs/testnet/errors) for a list of error codes and messages.

**Event format**

User Data Stream events for non-SBE sessions are sent as JSON in **text frames**, one event per frame.

Events in SBE sessions will be sent as **binary frames**.

Please refer to [userDataStream.subscribe](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#user-data-stream-subscribe) for details on how to subscribe to User Data Stream in WebSocket API.

Example of an event:

{  
 "event": {  
 "e": "outboundAccountPosition",  
 "E": 1728972148778,  
 "u": 1728972148778,  
 "B": [  
 {  
 "a": "ABC",  
 "f": "11818.00000000",  
 "l": "182.00000000"  
 },  
 {  
 "a": "DEF",  
 "f": "10580.00000000",  
 "l": "70.00000000"  
 }  
 ]  
 }  
}

Event fields:

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| event | OBJECT | YES | Event payload. See [User Data Streams](https://developers.binance.com/docs/binance-spot-api-docs/testnet/user-data-stream) |

**Rate limits**

**Connection limits**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#connection-limits)

There is a limit of **300 connections per attempt every 5 minutes**.

The connection is per **IP address**.

**General information on rate limits**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#general-information-on-rate-limits)

* Current API rate limits can be queried using the [exchangeInfo](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#exchange-information) request.
* There are multiple rate limit types across multiple intervals.
* Responses can indicate current rate limit status in the optional rateLimits field.
* Requests fail with status 429 when unfilled order count or request rate limits are violated.

**How to interpret rate limits**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#how-to-interpret-rate-limits)

A response with rate limit status may look like this:

{  
 "id": "7069b743-f477-4ae3-81db-db9b8df085d2",  
 "status": 200,  
 "result": {  
 "serverTime": 1656400526260  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 70  
 }  
 ]  
}

The rateLimits array describes all currently active rate limits affected by the request.

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| rateLimitType | ENUM | YES | Rate limit type: REQUEST\_WEIGHT, ORDERS |
| interval | ENUM | YES | Rate limit interval: SECOND, MINUTE, HOUR, DAY |
| intervalNum | INT | YES | Rate limit interval multiplier |
| limit | INT | YES | Request limit per interval |
| count | INT | YES | Current usage per interval |

Rate limits are accounted by intervals.

For example, a 1 MINUTE interval starts every minute. Request submitted at 00:01:23.456 counts towards the 00:01:00 minute's limit. Once the 00:02:00 minute starts, the count will reset to zero again.

Other intervals behave in a similar manner. For example, 1 DAY rate limit resets at 00:00 UTC every day, and 10 SECOND interval resets at 00, 10, 20... seconds of each minute.

APIs have multiple rate-limiting intervals. If you exhaust a shorter interval but the longer interval still allows requests, you will have to wait for the shorter interval to expire and reset. If you exhaust a longer interval, you will have to wait for that interval to reset, even if shorter rate limit count is zero.

**How to show/hide rate limit information**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#how-to-showhide-rate-limit-information)

rateLimits field is included with every response by default.

However, rate limit information can be quite bulky. If you are not interested in detailed rate limit status of every request, the rateLimits field can be omitted from responses to reduce their size.

* Optional returnRateLimits boolean parameter in request.

Use returnRateLimits parameter to control whether to include rateLimits fields in response to individual requests.

Default request and response:

{"id":1,"method":"time"}

{"id":1,"status":200,"result":{"serverTime":1656400526260},"rateLimits":[{"rateLimitType":"REQUEST\_WEIGHT","interval":"MINUTE","intervalNum":1,"limit":6000,"count":70}]}

Request and response without rate limit status:

{"id":2,"method":"time","params":{"returnRateLimits":false}}

{"id":2,"status":200,"result":{"serverTime":1656400527891}}

* Optional returnRateLimits boolean parameter in connection URL.

If you wish to omit rateLimits from all responses by default, use returnRateLimits parameter in the query string instead:

wss://ws-api.binance.com:443/ws-api/v3?returnRateLimits=false

This will make all requests made through this connection behave as if you have passed "returnRateLimits": false.

If you *want* to see rate limits for a particular request, you need to explicitly pass the "returnRateLimits": true parameter.

**Note:** Your requests are still rate limited if you hide the rateLimits field in responses.

**IP limits**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#ip-limits)

* Every request has a certain **weight**, added to your limit as you perform requests.
  + The heavier the request (e.g. querying data from multiple symbols), the more weight the request will cost.
  + Connecting to WebSocket API costs 2 weight.
* Current weight usage is indicated by the REQUEST\_WEIGHT rate limit type.
* Use the [exchangeInfo](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#exchange-information) request to keep track of the current weight limits.
* Weight is accumulated **per IP address** and is shared by all connections from that address.
* If you go over the weight limit, requests fail with status 429.
  + This status code indicates you should back off and stop spamming the API.
  + Rate-limited responses include a retryAfter field, indicating when you can retry the request.
* **Repeatedly violating rate limits and/or failing to back off after receiving 429s will result in an automated IP ban and you will be disconnected.**
  + Requests from a banned IP address fail with status 418.
  + retryAfter field indicates the timestamp when the ban will be lifted.
* IP bans are tracked and **scale in duration** for repeat offenders, **from 2 minutes to 3 days**.

Successful response indicating that in 1 minute you have used 70 weight out of your 6000 limit:

{  
 "id": "7069b743-f477-4ae3-81db-db9b8df085d2",  
 "status": 200,  
 "result": [],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 70  
 }  
 ]  
}

Failed response indicating that you are banned and the ban will last until epoch 1659146400000:

{  
 "id": "fc93a61a-a192-4cf4-bb2a-a8f0f0c51e06",  
 "status": 418,  
 "error": {  
 "code": -1003,  
 "msg": "Way too much request weight used; IP banned until 1659146400000. Please use WebSocket Streams for live updates to avoid bans.",  
 "data": {  
 "serverTime": 1659142907531,  
 "retryAfter": 1659146400000  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2411  
 }  
 ]  
}

**Unfilled Order Count**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/rate-limits#unfilled-order-count)

* Successfully placed orders update the ORDERS rate limit type.
* Rejected or unsuccessful orders might or might not update the ORDERS rate limit type.
* **Please note that if your orders are consistently filled by trades, you can continuously place orders on the API**. For more information, please see [Spot Unfilled Order Count Rules](https://developers.binance.com/docs/binance-spot-api-docs/faqs/order_count_decrement).
* Use the [account.rateLimits.orders](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-unfilled-order-count) request to keep track of how many orders you have placed within this interval.
* If you exceed this, requests fail with status 429.
  + This status code indicates you should back off and stop spamming the API.
  + Responses that have a status 429 include a retryAfter field, indicating when you can retry the request.
* This is maintained **per account** and is shared by all API keys of the account.

Successful response indicating that you have placed 12 orders in 10 seconds, and 4043 orders in the past 24 hours:

{  
 "id": "e2a85d9f-07a5-4f94-8d5f-789dc3deb097",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "orderId": 12510053279,  
 "orderListId": -1,  
 "clientOrderId": "a097fe6304b20a7e4fc436",  
 "transactTime": 1655716096505,  
 "price": "0.10000000",  
 "origQty": "10.00000000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "BUY",  
 "workingTime": 1655716096505,  
 "selfTradePreventionMode": "NONE"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 12  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 4043  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 321  
 }  
 ]  
}

**Request security**

* Every method has a security type which determines how to call it.
  + Security type is stated next to the method name. For example, [Place new order (TRADE)](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#place-new-order-trade).
  + If no security type is stated, the security type is NONE.

| Security type | API key | Signature | Description |
| --- | --- | --- | --- |
| NONE |  |  | Public market data |
| TRADE | required | required | Trading on the exchange, placing and canceling orders |
| USER\_DATA | required | required | Private account information, such as order status and your trading history |
| USER\_STREAM | required |  | Managing User Data Stream subscriptions |

* Secure methods require a valid API key to be specified and authenticated.
  + API keys can be created on the [API Management](https://www.binance.com/en/support/faq/360002502072) page of your Binance account.
  + **Both API key and secret key are sensitive.** Never share them with anyone. If you notice unusual activity in your account, immediately revoke all the keys and contact Binance support.
* API keys can be configured to allow access only to certain types of secure methods.
  + For example, you can have an API key with TRADE permission for trading, while using a separate API key with USER\_DATA permission to monitor your order status.
  + By default, an API key cannot TRADE. You need to enable trading in API Management first.
* TRADE and USER\_DATA requests are also known as SIGNED requests.

**SIGNED (TRADE and USER\_DATA) request security**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-trade-and-user_data-request-security)

* SIGNED requests require an additional parameter: signature, authorizing the request.
* Please consult [SIGNED request example (HMAC)](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-request-example-hmac), [SIGNED request example (RSA)](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-request-example-rsa), and [SIGNED request example (Ed25519)](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-request-example-ed25519) on how to compute signature, depending on which API key type you are using.

**Timing security**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#timing-security)

* SIGNED requests also require a timestamp parameter which should be the current timestamp either in milliseconds or microseconds. (See [General API Information](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#general-api-information))
* An additional optional parameter, recvWindow, specifies for how long the request stays valid and may only be specified in milliseconds.
  + If recvWindow is not sent, **it defaults to 5000 milliseconds**.
  + Maximum recvWindow is 60000 milliseconds.
* Request processing logic is as follows:

serverTime = getCurrentTime()  
if (timestamp < (serverTime + 1 second) && (serverTime - timestamp) <= recvWindow) {  
 *// begin processing request*  
 serverTime = getCurrentTime()  
 if (serverTime - timestamp) <= recvWindow {  
 *// forward request to Matching Engine*  
 } else {  
 *// reject request*  
 }  
 *// finish processing request*  
} else {  
 *// reject request*  
}

**Serious trading is about timing.** Networks can be unstable and unreliable, which can lead to requests taking varying amounts of time to reach the servers. With recvWindow, you can specify that the request must be processed within a certain number of milliseconds or be rejected by the server.

**It is recommended to use a small recvWindow of 5000 or less!**

**SIGNED request example (HMAC)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-request-example-hmac)

Here is a step-by-step guide on how to sign requests using HMAC secret key.

Example API key and secret key:

| Key | Value |
| --- | --- |
| apiKey | vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A |
| secretKey | NhqPtmdSJYdKjVHjA7PZj4Mge3R5YNiP1e3UZjInClVN65XAbvqqM6A7H5fATj0j |

**WARNING: DO NOT SHARE YOUR API KEY AND SECRET KEY WITH ANYONE.**

The example keys are provided here only for illustrative purposes.

Example of request:

{  
 "id": "4885f793-e5ad-4c3b-8f6c-55d891472b71",  
 "method": "order.place",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "quantity": "0.01000000",  
 "price": "52000.00",  
 "newOrderRespType": "ACK",  
 "recvWindow": 100,  
 "timestamp": 1645423376532,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "------ FILL ME ------"  
 }  
}

As you can see, the signature parameter is currently missing.

**Step 1. Construct the signature payload**

Take all request params except for the signature, sort them by name in alphabetical order:

| Parameter | Value |
| --- | --- |
| apiKey | vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A |
| newOrderRespType | ACK |
| price | 52000.00 |
| quantity | 0.01000000 |
| recvWindow | 100 |
| side | SELL |
| symbol | BTCUSDT |
| timeInForce | GTC |
| timestamp | 1645423376532 |
| type | LIMIT |

Format parameters as parameter=value pairs separated by &.

Resulting signature payload:

apiKey=vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A&newOrderRespType=ACK&price=52000.00&quantity=0.01000000&recvWindow=100&side=SELL&symbol=BTCUSDT&timeInForce=GTC&timestamp=1645423376532&type=LIMIT

**Step 2. Compute the signature**

1. Interpret secretKey as ASCII data, using it as a key for HMAC-SHA-256.
2. Sign signature payload as ASCII data.
3. Encode HMAC-SHA-256 output as a hex string.

Note that apiKey, secretKey, and the payload are **case-sensitive**, while resulting signature value is case-insensitive.

You can cross-check your signature algorithm implementation with OpenSSL:

$ echo -n 'apiKey=vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A&newOrderRespType=ACK&price=52000.00&quantity=0.01000000&recvWindow=100&side=SELL&symbol=BTCUSDT&timeInForce=GTC&timestamp=1645423376532&type=LIMIT' \  
 | openssl dgst -hex -sha256 -hmac 'NhqPtmdSJYdKjVHjA7PZj4Mge3R5YNiP1e3UZjInClVN65XAbvqqM6A7H5fATj0j'  
  
cc15477742bd704c29492d96c7ead9414dfd8e0ec4a00f947bb5bb454ddbd08a

**Step 3. Add signature to request params**

Finally, complete the request by adding the signature parameter with the signature string.

{  
 "id": "4885f793-e5ad-4c3b-8f6c-55d891472b71",  
 "method": "order.place",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "quantity": "0.01000000",  
 "price": "52000.00",  
 "newOrderRespType": "ACK",  
 "recvWindow": 100,  
 "timestamp": 1645423376532,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "cc15477742bd704c29492d96c7ead9414dfd8e0ec4a00f947bb5bb454ddbd08a"  
 }  
}

**SIGNED request example (RSA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-request-example-rsa)

Here is a step-by-step guide on how to sign requests using your RSA private key.

| Key | Value |
| --- | --- |
| apiKey | CAvIjXy3F44yW6Pou5k8Dy1swsYDWJZLeoK2r8G4cFDnE9nosRppc2eKc1T8TRTQ |

In this example, we assume the private key is stored in the test-prv-key.pem file.

**WARNING: DO NOT SHARE YOUR API KEY AND PRIVATE KEY WITH ANYONE.**

The example keys are provided here only for illustrative purposes.

Example of request:

{  
 "id": "4885f793-e5ad-4c3b-8f6c-55d891472b71",  
 "method": "order.place",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "quantity": "0.01000000",  
 "price": "52000.00",  
 "newOrderRespType": "ACK",  
 "recvWindow": 100,  
 "timestamp": 1645423376532,  
 "apiKey": "CAvIjXy3F44yW6Pou5k8Dy1swsYDWJZLeoK2r8G4cFDnE9nosRppc2eKc1T8TRTQ",  
 "signature": "------ FILL ME ------"  
 }  
}

**Step 1. Construct the signature payload**

Take all request params except for the signature, sort them by name in alphabetical order:

| Parameter | Value |
| --- | --- |
| apiKey | CAvIjXy3F44yW6Pou5k8Dy1swsYDWJZLeoK2r8G4cFDnE9nosRppc2eKc1T8TRTQ |
| newOrderRespType | ACK |
| price | 52000.00 |
| quantity | 0.01000000 |
| recvWindow | 100 |
| side | SELL |
| symbol | BTCUSDT |
| timeInForce | GTC |
| timestamp | 1645423376532 |
| type | LIMIT |

Format parameters as parameter=value pairs separated by &.

Resulting signature payload:

apiKey=CAvIjXy3F44yW6Pou5k8Dy1swsYDWJZLeoK2r8G4cFDnE9nosRppc2eKc1T8TRTQ&newOrderRespType=ACK&price=52000.00&quantity=0.01000000&recvWindow=100&side=SELL&symbol=BTCUSDT&timeInForce=GTC&timestamp=1645423376532&type=LIMIT

**Step 2. Compute the signature**

1. Encode signature payload as ASCII data.
2. Sign payload using RSASSA-PKCS1-v1\_5 algorithm with SHA-256 hash function.
3. Encode output as base64 string.

Note that apiKey, the payload, and the resulting signature are **case-sensitive**.

You can cross-check your signature algorithm implementation with OpenSSL:

$ echo -n 'apiKey=CAvIjXy3F44yW6Pou5k8Dy1swsYDWJZLeoK2r8G4cFDnE9nosRppc2eKc1T8TRTQ&newOrderRespType=ACK&price=52000.00&quantity=0.01000000&recvWindow=100&side=SELL&symbol=BTCUSDT&timeInForce=GTC&timestamp=1645423376532&type=LIMIT' \  
 | openssl dgst -sha256 -sign test-prv-key.pem \  
 | openssl enc -base64 -A  
  
OJJaf8C/3VGrU4ATTR4GiUDqL2FboSE1Qw7UnnoYNfXTXHubIl1iaePGuGyfct4NPu5oVEZCH4Q6ZStfB1w4ssgu0uiB/Bg+fBrRFfVgVaLKBdYHMvT+ljUJzqVaeoThG9oXlduiw8PbS9U8DYAbDvWN3jqZLo4Z2YJbyovyDAvDTr/oC0+vssLqP7NmlNb3fF3Bj7StmOwJvQJTbRAtzxK5PP7OQe+0mbW+D7RqVkUiSswR8qJFWTeSe4nXXNIdZdueYhF/Xf25L+KitJS5IHdIHcKfEw3MQzHFb2ZsGWkjDQwxkwr7Noi0Zaa+gFtxCuatGFm9dFIyx217pmSHtA==

**Step 3. Add signature to request params**

Finally, complete the request by adding the signature parameter with the signature string.

{  
 "id": "4885f793-e5ad-4c3b-8f6c-55d891472b71",  
 "method": "order.place",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "quantity": "0.01000000",  
 "price": "52000.00",  
 "newOrderRespType": "ACK",  
 "recvWindow": 100,  
 "timestamp": 1645423376532,  
 "apiKey": "CAvIjXy3F44yW6Pou5k8Dy1swsYDWJZLeoK2r8G4cFDnE9nosRppc2eKc1T8TRTQ",  
 "signature": "OJJaf8C/3VGrU4ATTR4GiUDqL2FboSE1Qw7UnnoYNfXTXHubIl1iaePGuGyfct4NPu5oVEZCH4Q6ZStfB1w4ssgu0uiB/Bg+fBrRFfVgVaLKBdYHMvT+ljUJzqVaeoThG9oXlduiw8PbS9U8DYAbDvWN3jqZLo4Z2YJbyovyDAvDTr/oC0+vssLqP7NmlNb3fF3Bj7StmOwJvQJTbRAtzxK5PP7OQe+0mbW+D7RqVkUiSswR8qJFWTeSe4nXXNIdZdueYhF/Xf25L+KitJS5IHdIHcKfEw3MQzHFb2ZsGWkjDQwxkwr7Noi0Zaa+gFtxCuatGFm9dFIyx217pmSHtA=="  
 }  
}

**SIGNED Request Example (Ed25519)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/request-security#signed-request-example-ed25519)

**Note: It is highly recommended to use Ed25519 API keys as it should provide the best performance and security out of all supported key types.**

| Parameter | Value |
| --- | --- |
| symbol | BTCUSDT |
| side | SELL |
| type | LIMIT |
| timeInForce | GTC |
| quantity | 1 |
| price | 0.2 |
| timestamp | 1668481559918 |

This is a sample code in Python to show how to sign the payload with an Ed25519 key.

*#!/usr/bin/env python3*  
  
import base64  
import time  
import json  
from cryptography.hazmat.primitives.serialization import load\_pem\_private\_key  
from websocket import create\_connection  
  
*# Set up authentication*  
API\_KEY='put your own API Key here'  
PRIVATE\_KEY\_PATH='test-prv-key.pem'  
  
*# Load the private key.*  
*# In this example the key is expected to be stored without encryption,*  
*# but we recommend using a strong password for improved security.*  
with open(PRIVATE\_KEY\_PATH, 'rb') as f:  
 private\_key = load\_pem\_private\_key(data=f.read(),  
 password=None)  
  
*# Set up the request parameters*  
params = {  
 'apiKey': API\_KEY,  
 'symbol': 'BTCUSDT',  
 'side': 'SELL',  
 'type': 'LIMIT',  
 'timeInForce': 'GTC',  
 'quantity': '1.0000000',  
 'price': '0.20'  
}  
  
*# Timestamp the request*  
timestamp = int(time.time() \* 1000) *# UNIX timestamp in milliseconds*  
params['timestamp'] = timestamp  
  
*# Sign the request*  
payload = '&'.join([f'{param}={value}' for param, value in sorted(params.items())])  
  
signature = base64.b64encode(private\_key.sign(payload.encode('ASCII')))  
params['signature'] = signature.decode('ASCII')  
  
*# Send the request*  
request = {  
 'id': 'my\_new\_order',  
 'method': 'order.place',  
 'params': params  
}  
  
ws = create\_connection("wss://ws-api.binance.com:443/ws-api/v3")  
ws.send(json.dumps(request))  
result = ws.recv()  
ws.close()  
  
print(result)

**Session Authentication**

**Note:** Only *Ed25519* keys are supported for this feature.

If you do not want to specify apiKey and signature in each individual request, you can authenticate your API key for the active WebSocket session.

Once authenticated, you no longer have to specify apiKey and signature for those requests that need them. Requests will be performed on behalf of the account owning the authenticated API key.

**Note:** You still have to specify the timestamp parameter for SIGNED requests.

**Authenticate after connection**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/session-authentication#authenticate-after-connection)

You can authenticate an already established connection using session authentication requests:

* [session.logon](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/session-authentication#log-in-with-api-key-signed) – authenticate, or change the API key associated with the connection
* [session.status](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/session-authentication#query-session-status) – check connection status and the current API key
* [session.logout](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/session-authentication#log-out-of-the-session) – forget the API key associated with the connection

**Regarding API key revocation:**

If during an active session the API key becomes invalid for *any reason* (e.g. IP address is not whitelisted, API key was deleted, API key doesn't have correct permissions, etc), after the next request the session will be revoked with the following error message:

{  
 "id": null,  
 "status": 401,  
 "error": {  
 "code": -2015,  
 "msg": "Invalid API-key, IP, or permissions for action."  
 }  
}

**Authorize *ad hoc* requests**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/session-authentication#authorize-ad-hoc-requests)

Only one API key can be authenticated with the WebSocket connection. The authenticated API key is used by default for requests that require an apiKey parameter. However, you can always specify the apiKey and signature explicitly for individual requests, overriding the authenticated API key and using a different one to authorize a specific request.

For example, you might want to authenticate your USER\_DATA key to be used by default, but specify the TRADE key with an explicit signature when placing orders.

**Data sources**

* The API system is asynchronous. Some delay in the response is normal and expected.
* Each method has a data source indicating where the data is coming from, and thus how up-to-date it is.

| Data Source | Latency | Description |
| --- | --- | --- |
| Matching Engine | lowest | The Matching Engine produces the response directly |
| Memory | low | Data is fetched from API server's local or external memory cache |
| Database | moderate | Data is retrieved from the database |

* Some methods have more than one data source (e.g., Memory => Database).

This means that the API will look for the latest data in that order: first in the cache, then in the database.

**General requests**

**Test connectivity**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/general-requests#test-connectivity)

{  
 "id": "922bcc6e-9de8-440d-9e84-7c80933a8d0d",  
 "method": "ping"  
}

Test connectivity to the WebSocket API.

**Note:** You can use regular WebSocket ping frames to test connectivity as well, WebSocket API will respond with pong frames as soon as possible. ping request along with time is a safe way to test request-response handling in your application.

**Weight:** 1

**Parameters:** NONE

**Data Source:** Memory

**Response:**

{  
 "id": "922bcc6e-9de8-440d-9e84-7c80933a8d0d",  
 "status": 200,  
 "result": {},  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Check server time**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/general-requests#check-server-time)

{  
 "id": "187d3cb2-942d-484c-8271-4e2141bbadb1",  
 "method": "time"  
}

Test connectivity to the WebSocket API and get the current server time.

**Weight:** 1

**Parameters:** NONE

**Data Source:** Memory

**Response:**

{  
 "id": "187d3cb2-942d-484c-8271-4e2141bbadb1",  
 "status": 200,  
 "result": {  
 "serverTime": 1656400526260  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Exchange information**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/general-requests#exchange-information)

{  
 "id": "5494febb-d167-46a2-996d-70533eb4d976",  
 "method": "exchangeInfo",  
 "params": {  
 "symbols": ["BNBBTC"]  
 }  
}

Query current exchange trading rules, rate limits, and symbol information.

**Weight:** 20

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | NO | Describe a single symbol |
| symbols | ARRAY of STRING | Describe multiple symbols |
| permissions | ARRAY of STRING | Filter symbols by permissions |
| showPermissionSets | BOOLEAN | Controls whether the content of the permissionSets field is populated or not. Defaults to true. |
| symbolStatus | ENUM | Filters symbols that have this tradingStatus.  Valid values: TRADING, HALT, BREAK Cannot be used in combination with symbol or symbols |

Notes:

* Only one of symbol, symbols, permissions parameters can be specified.
* Without parameters, exchangeInfo displays all symbols with ["SPOT, "MARGIN", "LEVERAGED"] permissions.
  + In order to list *all* active symbols on the exchange, you need to explicitly request all permissions.
* permissions accepts either a list of permissions, or a single permission name. E.g. "SPOT".
* [Available Permissions](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#account-and-symbol-permissions)

**Examples of Symbol Permissions Interpretation from the Response:**

* [["A","B"]] means you may place an order if your account has either permission "A" **or** permission "B".
* [["A"],["B"]] means you can place an order if your account has permission "A" **and** permission "B".
* [["A"],["B","C"]] means you can place an order if your account has permission "A" **and** permission "B" or permission "C". (Inclusive or is applied here, not exclusive or, so your account may have both permission "B" and permission "C".)

**Data Source:** Memory

**Response:**

{  
 "id": "5494febb-d167-46a2-996d-70533eb4d976",  
 "status": 200,  
 "result": {  
 "timezone": "UTC",  
 "serverTime": 1655969291181,  
 *// Global rate limits. See "Rate limits" section.*  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT", *// Rate limit type: REQUEST\_WEIGHT, ORDERS, CONNECTIONS*  
 "interval": "MINUTE", *// Rate limit interval: SECOND, MINUTE, DAY*  
 "intervalNum": 1, *// Rate limit interval multiplier (i.e., "1 minute")*  
 "limit": 6000 *// Rate limit per interval*  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000  
 },  
 {  
 "rateLimitType": "CONNECTIONS",  
 "interval": "MINUTE",  
 "intervalNum": 5,  
 "limit": 300  
 }  
 ],  
 *// Exchange filters are explained on the "Filters" page:*  
 *// https://github.com/binance/binance-spot-api-docs/blob/master/filters.md*  
 *// All exchange filters are optional.*  
 "exchangeFilters": [],  
 "symbols": [  
 {  
 "symbol": "BNBBTC",  
 "status": "TRADING",  
 "baseAsset": "BNB",  
 "baseAssetPrecision": 8,  
 "quoteAsset": "BTC",  
 "quotePrecision": 8,  
 "quoteAssetPrecision": 8,  
 "baseCommissionPrecision": 8,  
 "quoteCommissionPrecision": 8,  
 "orderTypes": [  
 "LIMIT",  
 "LIMIT\_MAKER",  
 "MARKET",  
 "STOP\_LOSS\_LIMIT",  
 "TAKE\_PROFIT\_LIMIT"  
 ],  
 "icebergAllowed": true,  
 "ocoAllowed": true,  
 "otoAllowed": true,  
 "quoteOrderQtyMarketAllowed": true,  
 "allowTrailingStop": true,  
 "cancelReplaceAllowed": true,  
 "allowAmend":false,  
 "isSpotTradingAllowed": true,  
 "isMarginTradingAllowed": true,  
 *// Symbol filters are explained on the "Filters" page:*  
 *// https://github.com/binance/binance-spot-api-docs/blob/master/filters.md*  
 *// All symbol filters are optional.*  
 "filters": [  
 {  
 "filterType": "PRICE\_FILTER",  
 "minPrice": "0.00000100",  
 "maxPrice": "100000.00000000",  
 "tickSize": "0.00000100"  
 },  
 {  
 "filterType": "LOT\_SIZE",  
 "minQty": "0.00100000",  
 "maxQty": "100000.00000000",  
 "stepSize": "0.00100000"  
 }  
 ],  
 "permissions": [],  
 "permissionSets": [  
 [  
 "SPOT",  
 "MARGIN",  
 "TRD\_GRP\_004"  
 ]  
 ],  
 "defaultSelfTradePreventionMode": "NONE",  
 "allowedSelfTradePreventionModes": [  
 "NONE"  
 ]  
 }  
 ],  
 *// Optional field. Present only when SOR is available.*  
 *// https://github.com/binance/binance-spot-api-docs/blob/master/faqs/sor\_faq.md*  
 "sors": [  
 {  
 "baseAsset": "BTC",  
 "symbols": [  
 "BTCUSDT",  
 "BTCUSDC"  
 ]  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Market data requests**

**Order book**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#order-book)

{  
 "id": "51e2affb-0aba-4821-ba75-f2625006eb43",  
 "method": "depth",  
 "params": {  
 "symbol": "BNBBTC",  
 "limit": 5  
 }  
}

Get current order book.

Note that this request returns limited market depth.

If you need to continuously monitor order book updates, please consider using WebSocket Streams:

* [<symbol>@depth<levels>](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#partial-book-depth-streams)
* [<symbol>@depth](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#diff-depth-stream)

You can use depth request together with <symbol>@depth streams to [maintain a local order book](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#how-to-manage-a-local-order-book-correctly).

**Weight:** Adjusted based on the limit:

| Limit | Weight |
| --- | --- |
| 1–100 | 5 |
| 101–500 | 25 |
| 501–1000 | 50 |
| 1001–5000 | 250 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| limit | INT | NO | Default: 100; Maximum: 5000 |

**Data Source:** Memory

**Response:**

{  
 "id": "51e2affb-0aba-4821-ba75-f2625006eb43",  
 "status": 200,  
 "result": {  
 "lastUpdateId": 2731179239,  
 *// Bid levels are sorted from highest to lowest price.*  
 "bids": [  
 [  
 "0.01379900", *// Price*  
 "3.43200000" *// Quantity*  
 ],  
 [  
 "0.01379800",  
 "3.24300000"  
 ],  
 [  
 "0.01379700",  
 "10.45500000"  
 ],  
 [  
 "0.01379600",  
 "3.82100000"  
 ],  
 [  
 "0.01379500",  
 "10.26200000"  
 ]  
 ],  
 *// Ask levels are sorted from lowest to highest price.*  
 "asks": [  
 [  
 "0.01380000",  
 "5.91700000"  
 ],  
 [  
 "0.01380100",  
 "6.01400000"  
 ],  
 [  
 "0.01380200",  
 "0.26800000"  
 ],  
 [  
 "0.01380300",  
 "0.33800000"  
 ],  
 [  
 "0.01380400",  
 "0.26800000"  
 ]  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Recent trades**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#recent-trades)

{  
 "id": "409a20bd-253d-41db-a6dd-687862a5882f",  
 "method": "trades.recent",  
 "params": {  
 "symbol": "BNBBTC",  
 "limit": 1  
 }  
}

Get recent trades.

If you need access to real-time trading activity, please consider using WebSocket Streams:

* [<symbol>@trade](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#trade-streams)

**Weight:** 25

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| limit | INT | NO | Default: 500; Maximum: 1000 |

**Data Source:** Memory

**Response:**

{  
 "id": "409a20bd-253d-41db-a6dd-687862a5882f",  
 "status": 200,  
 "result": [  
 {  
 "id": 194686783,  
 "price": "0.01361000",  
 "qty": "0.01400000",  
 "quoteQty": "0.00019054",  
 "time": 1660009530807,  
 "isBuyerMaker": true,  
 "isBestMatch": true  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Historical trades**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#historical-trades)

{  
 "id": "cffc9c7d-4efc-4ce0-b587-6b87448f052a",  
 "method": "trades.historical",  
 "params": {  
 "symbol": "BNBBTC",  
 "fromId": 0,  
 "limit": 1  
 }  
}

Get historical trades.

**Weight:** 25

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| fromId | INT | NO | Trade ID to begin at |
| limit | INT | NO | Default: 500; Maximum: 1000 |

Notes:

* If fromId is not specified, the most recent trades are returned.

**Data Source:** Database

**Response:**

{  
 "id": "cffc9c7d-4efc-4ce0-b587-6b87448f052a",  
 "status": 200,  
 "result": [  
 {  
 "id": 0,  
 "price": "0.00005000",  
 "qty": "40.00000000",  
 "quoteQty": "0.00200000",  
 "time": 1500004800376,  
 "isBuyerMaker": true,  
 "isBestMatch": true  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 10  
 }  
 ]  
}

**Aggregate trades**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#aggregate-trades)

{  
 "id": "189da436-d4bd-48ca-9f95-9f613d621717",  
 "method": "trades.aggregate",  
 "params": {  
 "symbol": "BNBBTC",  
 "fromId": 50000000,  
 "limit": 1  
 }  
}

Get aggregate trades.

An *aggregate trade* (aggtrade) represents one or more individual trades. Trades that fill at the same time, from the same taker order, with the same price – those trades are collected into an aggregate trade with total quantity of the individual trades.

If you need access to real-time trading activity, please consider using WebSocket Streams:

* [<symbol>@aggTrade](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#aggregate-trade-streams)

If you need historical aggregate trade data, please consider using [data.binance.vision](https://github.com/binance/binance-public-data/#aggtrades).

**Weight:** 4

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| fromId | INT | NO | Aggregate trade ID to begin at |
| startTime | LONG | NO |  |
| endTime | LONG | NO |  |
| limit | INT | NO | Default: 500; Maximum: 1000 |

Notes:

* If fromId is specified, return aggtrades with aggregate trade ID >= fromId.

Use fromId and limit to page through all aggtrades.

* If startTime and/or endTime are specified, aggtrades are filtered by execution time (T).

fromId cannot be used together with startTime and endTime.

* If no condition is specified, the most recent aggregate trades are returned.

**Data Source:** Database

**Response:**

{  
 "id": "189da436-d4bd-48ca-9f95-9f613d621717",  
 "status": 200,  
 "result": [  
 {  
 "a": 50000000, *// Aggregate trade ID*  
 "p": "0.00274100", *// Price*  
 "q": "57.19000000", *// Quantity*  
 "f": 59120167, *// First trade ID*  
 "l": 59120170, *// Last trade ID*  
 "T": 1565877971222, *// Timestamp*  
 "m": true, *// Was the buyer the maker?*  
 "M": true *// Was the trade the best price match?*  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Klines**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#klines)

{  
 "id": "1dbbeb56-8eea-466a-8f6e-86bdcfa2fc0b",  
 "method": "klines",  
 "params": {  
 "symbol": "BNBBTC",  
 "interval": "1h",  
 "startTime": 1655969280000,  
 "limit": 1  
 }  
}

Get klines (candlestick bars).

Klines are uniquely identified by their open & close time.

If you need access to real-time kline updates, please consider using WebSocket Streams:

* [<symbol>@kline\_<interval>](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#klinecandlestick-streams)

If you need historical kline data, please consider using [data.binance.vision](https://github.com/binance/binance-public-data/#klines).

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| interval | ENUM | YES |  |
| startTime | LONG | NO |  |
| endTime | LONG | NO |  |
| timeZone | STRING | NO | Default: 0 (UTC) |
| limit | INT | NO | Default: 500; Maximum: 1000 |

Supported kline intervals (case-sensitive):

| Interval | interval value |
| --- | --- |
| seconds | 1s |
| minutes | 1m, 3m, 5m, 15m, 30m |
| hours | 1h, 2h, 4h, 6h, 8h, 12h |
| days | 1d, 3d |
| weeks | 1w |
| months | 1M |

Notes:

* If startTime, endTime are not specified, the most recent klines are returned.
* Supported values for timeZone:
  + Hours and minutes (e.g. -1:00, 05:45)
  + Only hours (e.g. 0, 8, 4)
  + Accepted range is strictly [-12:00 to +14:00] inclusive
* If timeZone provided, kline intervals are interpreted in that timezone instead of UTC.
* Note that startTime and endTime are always interpreted in UTC, regardless of timeZone.

**Data Source:** Database

**Response:**

{  
 "id": "1dbbeb56-8eea-466a-8f6e-86bdcfa2fc0b",  
 "status": 200,  
 "result": [  
 [  
 1655971200000, *// Kline open time*  
 "0.01086000", *// Open price*  
 "0.01086600", *// High price*  
 "0.01083600", *// Low price*  
 "0.01083800", *// Close price*  
 "2290.53800000", *// Volume*  
 1655974799999, *// Kline close time*  
 "24.85074442", *// Quote asset volume*  
 2283, *// Number of trades*  
 "1171.64000000", *// Taker buy base asset volume*  
 "12.71225884", *// Taker buy quote asset volume*  
 "0" *// Unused field, ignore*  
 ]  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**UI Klines**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#ui-klines)

{  
 "id": "b137468a-fb20-4c06-bd6b-625148eec958",  
 "method": "uiKlines",  
 "params": {  
 "symbol": "BNBBTC",  
 "interval": "1h",  
 "startTime": 1655969280000,  
 "limit": 1  
 }  
}

Get klines (candlestick bars) optimized for presentation.

This request is similar to [klines](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#klines), having the same parameters and response. uiKlines return modified kline data, optimized for presentation of candlestick charts.

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| interval | ENUM | YES | See [klines](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#kline-intervals) |
| startTime | LONG | NO |  |
| endTime | LONG | NO |  |
| timeZone | STRING | NO | Default: 0 (UTC) |
| limit | INT | NO | Default: 500; Maximum: 1000 |

Notes:

* If startTime, endTime are not specified, the most recent klines are returned.
* Supported values for timeZone:
  + Hours and minutes (e.g. -1:00, 05:45)
  + Only hours (e.g. 0, 8, 4)
  + Accepted range is strictly [-12:00 to +14:00] inclusive
* If timeZone provided, kline intervals are interpreted in that timezone instead of UTC.
* Note that startTime and endTime are always interpreted in UTC, regardless of timeZone.

**Data Source:** Database

**Response:**

{  
 "id": "b137468a-fb20-4c06-bd6b-625148eec958",  
 "status": 200,  
 "result": [  
 [  
 1655971200000, *// Kline open time*  
 "0.01086000", *// Open price*  
 "0.01086600", *// High price*  
 "0.01083600", *// Low price*  
 "0.01083800", *// Close price*  
 "2290.53800000", *// Volume*  
 1655974799999, *// Kline close time*  
 "24.85074442", *// Quote asset volume*  
 2283, *// Number of trades*  
 "1171.64000000", *// Taker buy base asset volume*  
 "12.71225884", *// Taker buy quote asset volume*  
 "0" *// Unused field, ignore*  
 ]  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Current average price**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#current-average-price)

{  
 "id": "ddbfb65f-9ebf-42ec-8240-8f0f91de0867",  
 "method": "avgPrice",  
 "params": {  
 "symbol": "BNBBTC"  
 }  
}

Get current average price for a symbol.

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |

**Data Source:** Memory

**Response:**

{  
 "id": "ddbfb65f-9ebf-42ec-8240-8f0f91de0867",  
 "status": 200,  
 "result": {  
 "mins": 5, *// Average price interval (in minutes)*  
 "price": "9.35751834", *// Average price*  
 "closeTime": 1694061154503 *// Last trade time*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**24hr ticker price change statistics**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#24hr-ticker-price-change-statistics)

{  
 "id": "93fb61ef-89f8-4d6e-b022-4f035a3fadad",  
 "method": "ticker.24hr",  
 "params": {  
 "symbol": "BNBBTC"  
 }  
}

Get 24-hour rolling window price change statistics.

If you need to continuously monitor trading statistics, please consider using WebSocket Streams:

* [<symbol>@ticker](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-ticker-streams) or [!ticker@arr](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#all-market-tickers-stream)
* [<symbol>@miniTicker](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-mini-ticker-stream) or [!miniTicker@arr](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#all-market-mini-tickers-stream)

If you need different window sizes, use the [ticker](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#rolling-window-price-change-statistics) request.

**Weight:** Adjusted based on the number of requested symbols:

| Symbols | Weight |
| --- | --- |
| 1–20 | 2 |
| 21–100 | 40 |
| 101 or more | 80 |
| all symbols | 80 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | NO | Query ticker for a single symbol |
| symbols | ARRAY of STRING | Query ticker for multiple symbols |
| type | ENUM | NO | Ticker type: FULL (default) or MINI |

Notes:

* symbol and symbols cannot be used together.
* If no symbol is specified, returns information about all symbols currently trading on the exchange.

**Data Source:** Memory

**Response:**

FULL type, for a single symbol:

{  
 "id": "93fb61ef-89f8-4d6e-b022-4f035a3fadad",  
 "status": 200,  
 "result": {  
 "symbol": "BNBBTC",  
 "priceChange": "0.00013900",  
 "priceChangePercent": "1.020",  
 "weightedAvgPrice": "0.01382453",  
 "prevClosePrice": "0.01362800",  
 "lastPrice": "0.01376700",  
 "lastQty": "1.78800000",  
 "bidPrice": "0.01376700",  
 "bidQty": "4.64600000",  
 "askPrice": "0.01376800",  
 "askQty": "14.31400000",  
 "openPrice": "0.01362800",  
 "highPrice": "0.01414900",  
 "lowPrice": "0.01346600",  
 "volume": "69412.40500000",  
 "quoteVolume": "959.59411487",  
 "openTime": 1660014164909,  
 "closeTime": 1660100564909,  
 "firstId": 194696115, *// First trade ID*  
 "lastId": 194968287, *// Last trade ID*  
 "count": 272173 *// Number of trades*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

MINI type, for a single symbol:

{  
 "id": "9fa2a91b-3fca-4ed7-a9ad-58e3b67483de",  
 "status": 200,  
 "result": {  
 "symbol": "BNBBTC",  
 "openPrice": "0.01362800",  
 "highPrice": "0.01414900",  
 "lowPrice": "0.01346600",  
 "lastPrice": "0.01376700",  
 "volume": "69412.40500000",  
 "quoteVolume": "959.59411487",  
 "openTime": 1660014164909,  
 "closeTime": 1660100564909,  
 "firstId": 194696115, *// First trade ID*  
 "lastId": 194968287, *// Last trade ID*  
 "count": 272173 *// Number of trades*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

If more than one symbol is requested, response returns an array:

{  
 "id": "901be0d9-fd3b-45e4-acd6-10c580d03430",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BNBBTC",  
 "priceChange": "0.00016500",  
 "priceChangePercent": "1.213",  
 "weightedAvgPrice": "0.01382508",  
 "prevClosePrice": "0.01360800",  
 "lastPrice": "0.01377200",  
 "lastQty": "1.01400000",  
 "bidPrice": "0.01377100",  
 "bidQty": "7.55700000",  
 "askPrice": "0.01377200",  
 "askQty": "4.37900000",  
 "openPrice": "0.01360700",  
 "highPrice": "0.01414900",  
 "lowPrice": "0.01346600",  
 "volume": "69376.27900000",  
 "quoteVolume": "959.13277091",  
 "openTime": 1660014615517,  
 "closeTime": 1660101015517,  
 "firstId": 194697254,  
 "lastId": 194969483,  
 "count": 272230  
 },  
 {  
 "symbol": "BTCUSDT",  
 "priceChange": "-938.06000000",  
 "priceChangePercent": "-3.938",  
 "weightedAvgPrice": "23265.34432003",  
 "prevClosePrice": "23819.17000000",  
 "lastPrice": "22880.91000000",  
 "lastQty": "0.00536000",  
 "bidPrice": "22880.40000000",  
 "bidQty": "0.00424000",  
 "askPrice": "22880.91000000",  
 "askQty": "0.04276000",  
 "openPrice": "23818.97000000",  
 "highPrice": "23933.25000000",  
 "lowPrice": "22664.69000000",  
 "volume": "153508.37606000",  
 "quoteVolume": "3571425225.04441220",  
 "openTime": 1660014615977,  
 "closeTime": 1660101015977,  
 "firstId": 1592019902,  
 "lastId": 1597301762,  
 "count": 5281861  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Trading Day Ticker**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#trading-day-ticker)

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "method": "ticker.tradingDay",  
 "params": {  
 "symbols": [  
 "BNBBTC",  
 "BTCUSDT"  
 ],  
 "timeZone": "00:00"  
 }  
}

Price change statistics for a trading day.

**Weight:**

4 for each requested symbol.  
  
The weight for this request will cap at 200 once the number of symbols in the request is more than 50.

**Parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| symbol | STRING | YES | Query ticker of a single symbol |
| symbols | ARRAY of STRING | Query ticker for multiple symbols |
| timeZone | STRING | NO | Default: 0 (UTC) |
| type | ENUM | NO | Supported values: FULL or MINI. If none provided, the default is FULL |

**Notes:**

* Supported values for timeZone:
  + Hours and minutes (e.g. -1:00, 05:45)
  + Only hours (e.g. 0, 8, 4)

**Data Source:** Database

**Response: - FULL**

With symbol:

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "priceChange": "-83.13000000", *// Absolute price change*  
 "priceChangePercent": "-0.317", *// Relative price change in percent*  
 "weightedAvgPrice": "26234.58803036", *// quoteVolume / volume*  
 "openPrice": "26304.80000000",  
 "highPrice": "26397.46000000",  
 "lowPrice": "26088.34000000",  
 "lastPrice": "26221.67000000",  
 "volume": "18495.35066000", *// Volume in base asset*  
 "quoteVolume": "485217905.04210480",  
 "openTime": 1695686400000,  
 "closeTime": 1695772799999,  
 "firstId": 3220151555,  
 "lastId": 3220849281,  
 "count": 697727  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

With symbols:

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "priceChange": "-83.13000000",  
 "priceChangePercent": "-0.317",  
 "weightedAvgPrice": "26234.58803036",  
 "openPrice": "26304.80000000",  
 "highPrice": "26397.46000000",  
 "lowPrice": "26088.34000000",  
 "lastPrice": "26221.67000000",  
 "volume": "18495.35066000",  
 "quoteVolume": "485217905.04210480",  
 "openTime": 1695686400000,  
 "closeTime": 1695772799999,  
 "firstId": 3220151555,  
 "lastId": 3220849281,  
 "count": 697727  
 },  
 {  
 "symbol": "BNBUSDT",  
 "priceChange": "2.60000000",  
 "priceChangePercent": "1.238",  
 "weightedAvgPrice": "211.92276958",  
 "openPrice": "210.00000000",  
 "highPrice": "213.70000000",  
 "lowPrice": "209.70000000",  
 "lastPrice": "212.60000000",  
 "volume": "280709.58900000",  
 "quoteVolume": "59488753.54750000",  
 "openTime": 1695686400000,  
 "closeTime": 1695772799999,  
 "firstId": 672397461,  
 "lastId": 672496158,  
 "count": 98698  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 8  
 }  
 ]  
}

**Response: - MINI**

With symbol:

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "openPrice": "26304.80000000",  
 "highPrice": "26397.46000000",  
 "lowPrice": "26088.34000000",  
 "lastPrice": "26221.67000000",  
 "volume": "18495.35066000", *// Volume in base asset*  
 "quoteVolume": "485217905.04210480", *// Volume in quote asset*  
 "openTime": 1695686400000,  
 "closeTime": 1695772799999,  
 "firstId": 3220151555, *// Trade ID of the first trade in the interval*  
 "lastId": 3220849281, *// Trade ID of the last trade in the interval*  
 "count": 697727 *// Number of trades in the interval*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

With symbols:

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "openPrice": "26304.80000000",  
 "highPrice": "26397.46000000",  
 "lowPrice": "26088.34000000",  
 "lastPrice": "26221.67000000",  
 "volume": "18495.35066000",  
 "quoteVolume": "485217905.04210480",  
 "openTime": 1695686400000,  
 "closeTime": 1695772799999,  
 "firstId": 3220151555,  
 "lastId": 3220849281,  
 "count": 697727  
 },  
 {  
 "symbol": "BNBUSDT",  
 "openPrice": "210.00000000",  
 "highPrice": "213.70000000",  
 "lowPrice": "209.70000000",  
 "lastPrice": "212.60000000",  
 "volume": "280709.58900000",  
 "quoteVolume": "59488753.54750000",  
 "openTime": 1695686400000,  
 "closeTime": 1695772799999,  
 "firstId": 672397461,  
 "lastId": 672496158,  
 "count": 98698  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 8  
 }  
 ]  
}

**Rolling window price change statistics**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#rolling-window-price-change-statistics)

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "method": "ticker",  
 "params": {  
 "symbols": [  
 "BNBBTC",  
 "BTCUSDT"  
 ],  
 "windowSize": "7d"  
 }  
}

Get rolling window price change statistics with a custom window.

This request is similar to [ticker.24hr](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#24hr-ticker-price-change-statistics), but statistics are computed on demand using the arbitrary window you specify.

**Note:** Window size precision is limited to 1 minute. While the closeTime is the current time of the request, openTime always start on a minute boundary. As such, the effective window might be up to 59999 ms wider than the requested windowSize.

Window computation example

If you need to continuously monitor trading statistics, please consider using WebSocket Streams:

* [<symbol>@ticker\_<window\_size>](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-rolling-window-statistics-streams) or [!ticker\_<window-size>@arr](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#all-market-rolling-window-statistics-streams)

**Weight:** Adjusted based on the number of requested symbols:

| Symbols | Weight |
| --- | --- |
| 1–50 | 4 per symbol |
| 51–100 | 200 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES | Query ticker of a single symbol |
| symbols | ARRAY of STRING | Query ticker for multiple symbols |
| type | ENUM | NO | Ticker type: FULL (default) or MINI |
| windowSize | ENUM | NO | Default 1d |

Supported window sizes:

| Unit | windowSize value |
| --- | --- |
| minutes | 1m, 2m ... 59m |
| hours | 1h, 2h ... 23h |
| days | 1d, 2d ... 7d |

Notes:

* Either symbol or symbols must be specified.
* Maximum number of symbols in one request: 200.
* Window size units cannot be combined. E.g., 1d 2h is not supported.

**Data Source:** Database

**Response:**

FULL type, for a single symbol:

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "status": 200,  
 "result": {  
 "symbol": "BNBBTC",  
 "priceChange": "0.00061500",  
 "priceChangePercent": "4.735",  
 "weightedAvgPrice": "0.01368242",  
 "openPrice": "0.01298900",  
 "highPrice": "0.01418800",  
 "lowPrice": "0.01296000",  
 "lastPrice": "0.01360400",  
 "volume": "587179.23900000",  
 "quoteVolume": "8034.03382165",  
 "openTime": 1659580020000,  
 "closeTime": 1660184865291,  
 "firstId": 192977765, *// First trade ID*  
 "lastId": 195365758, *// Last trade ID*  
 "count": 2387994 *// Number of trades*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

MINI type, for a single symbol:

{  
 "id": "bdb7c503-542c-495c-b797-4d2ee2e91173",  
 "status": 200,  
 "result": {  
 "symbol": "BNBBTC",  
 "openPrice": "0.01298900",  
 "highPrice": "0.01418800",  
 "lowPrice": "0.01296000",  
 "lastPrice": "0.01360400",  
 "volume": "587179.23900000",  
 "quoteVolume": "8034.03382165",  
 "openTime": 1659580020000,  
 "closeTime": 1660184865291,  
 "firstId": 192977765, *// First trade ID*  
 "lastId": 195365758, *// Last trade ID*  
 "count": 2387994 *// Number of trades*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

If more than one symbol is requested, response returns an array:

{  
 "id": "f4b3b507-c8f2-442a-81a6-b2f12daa030f",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BNBBTC",  
 "priceChange": "0.00061500",  
 "priceChangePercent": "4.735",  
 "weightedAvgPrice": "0.01368242",  
 "openPrice": "0.01298900",  
 "highPrice": "0.01418800",  
 "lowPrice": "0.01296000",  
 "lastPrice": "0.01360400",  
 "volume": "587169.48600000",  
 "quoteVolume": "8033.90114517",  
 "openTime": 1659580020000,  
 "closeTime": 1660184820927,  
 "firstId": 192977765,  
 "lastId": 195365700,  
 "count": 2387936  
 },  
 {  
 "symbol": "BTCUSDT",  
 "priceChange": "1182.92000000",  
 "priceChangePercent": "5.113",  
 "weightedAvgPrice": "23349.27074846",  
 "openPrice": "23135.33000000",  
 "highPrice": "24491.22000000",  
 "lowPrice": "22400.00000000",  
 "lastPrice": "24318.25000000",  
 "volume": "1039498.10978000",  
 "quoteVolume": "24271522807.76838630",  
 "openTime": 1659580020000,  
 "closeTime": 1660184820927,  
 "firstId": 1568787779,  
 "lastId": 1604337406,  
 "count": 35549628  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 8  
 }  
 ]  
}

**Symbol price ticker**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#symbol-price-ticker)

{  
 "id": "043a7cf2-bde3-4888-9604-c8ac41fcba4d",  
 "method": "ticker.price",  
 "params": {  
 "symbol": "BNBBTC"  
 }  
}

Get the latest market price for a symbol.

If you need access to real-time price updates, please consider using WebSocket Streams:

* [<symbol>@aggTrade](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#aggregate-trade-streams)
* [<symbol>@trade](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#trade-streams)

**Weight:** Adjusted based on the number of requested symbols:

| Parameter | Weight |
| --- | --- |
| symbol | 2 |
| symbols | 4 |
| none | 4 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | NO | Query price for a single symbol |
| symbols | ARRAY of STRING | Query price for multiple symbols |

Notes:

* symbol and symbols cannot be used together.
* If no symbol is specified, returns information about all symbols currently trading on the exchange.

**Data Source:** Memory

**Response:**

{  
 "id": "043a7cf2-bde3-4888-9604-c8ac41fcba4d",  
 "status": 200,  
 "result": {  
 "symbol": "BNBBTC",  
 "price": "0.01361900"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

If more than one symbol is requested, response returns an array:

{  
 "id": "e739e673-24c8-4adf-9cfa-b81f30330b09",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BNBBTC",  
 "price": "0.01363700"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "price": "24267.15000000"  
 },  
 {  
 "symbol": "BNBBUSD",  
 "price": "331.10000000"  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

**Symbol order book ticker**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/market-data-requests#symbol-order-book-ticker)

{  
 "id": "057deb3a-2990-41d1-b58b-98ea0f09e1b4",  
 "method": "ticker.book",  
 "params": {  
 "symbols": [  
 "BNBBTC",  
 "BTCUSDT"  
 ]  
 }  
}

Get the current best price and quantity on the order book.

If you need access to real-time order book ticker updates, please consider using WebSocket Streams:

* [<symbol>@bookTicker](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-book-ticker-streams)

**Weight:** Adjusted based on the number of requested symbols:

| Parameter | Weight |
| --- | --- |
| symbol | 2 |
| symbols | 4 |
| none | 4 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | NO | Query ticker for a single symbol |
| symbols | ARRAY of STRING | Query ticker for multiple symbols |

Notes:

* symbol and symbols cannot be used together.
* If no symbol is specified, returns information about all symbols currently trading on the exchange.

**Data Source:** Memory

**Response:**

{  
 "id": "9d32157c-a556-4d27-9866-66760a174b57",  
 "status": 200,  
 "result": {  
 "symbol": "BNBBTC",  
 "bidPrice": "0.01358000",  
 "bidQty": "12.53400000",  
 "askPrice": "0.01358100",  
 "askQty": "17.83700000"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

If more than one symbol is requested, response returns an array:

{  
 "id": "057deb3a-2990-41d1-b58b-98ea0f09e1b4",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BNBBTC",  
 "bidPrice": "0.01358000",  
 "bidQty": "12.53400000",  
 "askPrice": "0.01358100",  
 "askQty": "17.83700000"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "bidPrice": "23980.49000000",  
 "bidQty": "0.01000000",  
 "askPrice": "23981.31000000",  
 "askQty": "0.01512000"  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

**Authentication requests**

**Note:** Only *Ed25519* keys are supported for this feature.

**Log in with API key (SIGNED)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/authentication-requests#log-in-with-api-key-signed)

{  
 "id": "c174a2b1-3f51-4580-b200-8528bd237cb7",  
 "method": "session.logon",  
 "params": {  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "1cf54395b336b0a9727ef27d5d98987962bc47aca6e13fe978612d0adee066ed",  
 "timestamp": 1649729878532  
 }  
}

Authenticate WebSocket connection using the provided API key.

After calling session.logon, you can omit apiKey and signature parameters for future requests that require them.

Note that only one API key can be authenticated. Calling session.logon multiple times changes the current authenticated API key.

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

**Data Source:** Memory

**Response:**

{  
 "id": "c174a2b1-3f51-4580-b200-8528bd237cb7",  
 "status": 200,  
 "result": {  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "authorizedSince": 1649729878532,  
 "connectedSince": 1649729873021,  
 "returnRateLimits": false,  
 "serverTime": 1649729878630,  
 "userDataStream": false *// is User Data Stream subscription active?*  
 }  
}

**Query session status**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/authentication-requests#query-session-status)

{  
 "id": "b50c16cd-62c9-4e29-89e4-37f10111f5bf",  
 "method": "session.status"  
}

Query the status of the WebSocket connection, inspecting which API key (if any) is used to authorize requests.

**Weight:** 2

**Parameters:** NONE

**Data Source:** Memory

**Response:**

{  
 "id": "b50c16cd-62c9-4e29-89e4-37f10111f5bf",  
 "status": 200,  
 "result": {  
 *// if the connection is not authenticated, "apiKey" and "authorizedSince" will be shown as null*  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "authorizedSince": 1649729878532,  
 "connectedSince": 1649729873021,  
 "returnRateLimits": false,  
 "serverTime": 1649730611671,  
 "userDataStream": true *// is User Data Stream subscription active?*  
 }  
}

**Log out of the session**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/authentication-requests#log-out-of-the-session)

{  
 "id": "c174a2b1-3f51-4580-b200-8528bd237cb7",  
 "method": "session.logout"  
}

Forget the API key previously authenticated. If the connection is not authenticated, this request does nothing.

Note that the WebSocket connection stays open after session.logout request. You can continue using the connection, but now you will have to explicitly provide the apiKey and signature parameters where needed.

**Weight:** 2

**Parameters:** NONE

**Data Source:** Memory

**Response:**

{  
 "id": "c174a2b1-3f51-4580-b200-8528bd237cb7",  
 "status": 200,  
 "result": {  
 "apiKey": null,  
 "authorizedSince": null,  
 "connectedSince": 1649729873021,  
 "returnRateLimits": false,  
 "serverTime": 1649730611671,  
 "userDataStream": false *// is User Data Stream subscription active?*  
 }  
}

**Trading requests**

**Place new order (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-trade)

{  
 "id": "56374a46-3061-486b-a311-99ee972eb648",  
 "method": "order.place",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "price": "23416.10000000",  
 "quantity": "0.00847000",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "15af09e41c36f3cc61378c2fbe2c33719a03dd5eba8d0f9206fbda44de717c88",  
 "timestamp": 1660801715431  
 }  
}

Send in a new order.

This adds 1 order to the EXCHANGE\_MAX\_ORDERS filter and the MAX\_NUM\_ORDERS filter.

**Weight:** 1

**Unfilled Order Count:** 1

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| side | ENUM | YES | BUY or SELL |
| type | ENUM | YES |  |
| timeInForce | ENUM | NO \* |  |
| price | DECIMAL | NO \* |  |
| quantity | DECIMAL | NO \* |  |
| quoteOrderQty | DECIMAL | NO \* |  |
| newClientOrderId | STRING | NO | Arbitrary unique ID among open orders. Automatically generated if not sent |
| newOrderRespType | ENUM | NO | Select response format: ACK, RESULT, FULL.  MARKET and LIMIT orders use FULL by default, other order types default to ACK. |
| stopPrice | DECIMAL | NO \* |  |
| trailingDelta | INT | NO \* | See [Trailing Stop order FAQ](https://developers.binance.com/docs/binance-spot-api-docs/faqs/trailing-stop-faq) |
| icebergQty | DECIMAL | NO |  |
| strategyId | LONG | NO | Arbitrary numeric value identifying the order within an order strategy. |
| strategyType | INT | NO | Arbitrary numeric value identifying the order strategy.  Values smaller than 1000000 are reserved and cannot be used. |
| selfTradePreventionMode | ENUM | NO | The allowed enums is dependent on what is configured on the symbol. Supported values: [STP Modes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#stpmodes) |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Certain parameters (\*) become mandatory based on the order type:

| Order type | Mandatory parameters |
| --- | --- |
| LIMIT | * timeInForce * price * quantity |
| LIMIT\_MAKER | * price * quantity |
| MARKET | * quantity or quoteOrderQty |
| STOP\_LOSS | * quantity * stopPrice or trailingDelta |
| STOP\_LOSS\_LIMIT | * timeInForce * price * quantity * stopPrice or trailingDelta |
| TAKE\_PROFIT | * quantity * stopPrice or trailingDelta |
| TAKE\_PROFIT\_LIMIT | * timeInForce * price * quantity * stopPrice or trailingDelta |

Supported order types:

| Order type | Description |
| --- | --- |
| LIMIT | Buy or sell quantity at the specified price or better. |
| LIMIT\_MAKER | LIMIT order that will be rejected if it immediately matches and trades as a taker.  This order type is also known as a POST-ONLY order. |
| MARKET | Buy or sell at the best available market price.   * MARKET order with quantity parameter specifies the amount of the *base asset* you want to buy or sell. Actually executed quantity of the quote asset will be determined by available market liquidity.   E.g., a MARKET BUY order on BTCUSDT for "quantity": "0.1000" specifies that you want to buy 0.1 BTC at the best available price. If there is not enough BTC at the best price, keep buying at the next best price, until either your order is filled, or you run out of USDT, or market runs out of BTC.   * MARKET order with quoteOrderQty parameter specifies the amount of the *quote asset* you want to spend (when buying) or receive (when selling). Actually executed quantity of the base asset will be determined by available market liquidity.   E.g., a MARKET BUY on BTCUSDT for "quoteOrderQty": "100.00" specifies that you want to buy as much BTC as you can for 100 USDT at the best available price. Similarly, a SELL order will sell as much available BTC as needed for you to receive 100 USDT (before commission). |
| STOP\_LOSS | Execute a MARKET order for given quantity when specified conditions are met.  I.e., when stopPrice is reached, or when trailingDelta is activated. |
| STOP\_LOSS\_LIMIT | Place a LIMIT order with given parameters when specified conditions are met. |
| TAKE\_PROFIT | Like STOP\_LOSS but activates when market price moves in the favorable direction. |
| TAKE\_PROFIT\_LIMIT | Like STOP\_LOSS\_LIMIT but activates when market price moves in the favorable direction. |

Available timeInForce options, setting how long the order should be active before expiration:

| TIF | Description |
| --- | --- |
| GTC | **Good 'til Canceled** – the order will remain on the book until you cancel it, or the order is completely filled. |
| IOC | **Immediate or Cancel** – the order will be filled for as much as possible, the unfilled quantity immediately expires. |
| FOK | **Fill or Kill** – the order will expire unless it cannot be immediately filled for the entire quantity. |

Notes:

* newClientOrderId specifies clientOrderId value for the order.

A new order with the same clientOrderId is accepted only when the previous one is filled or expired.

* Any LIMIT or LIMIT\_MAKER order can be made into an iceberg order by specifying the icebergQty.

An order with an icebergQty must have timeInForce set to GTC.

* Trigger order price rules for STOP\_LOSS/TAKE\_PROFIT orders:
  + stopPrice must be above market price: STOP\_LOSS BUY, TAKE\_PROFIT SELL
  + stopPrice must be below market price: STOP\_LOSS SELL, TAKE\_PROFIT BUY
* MARKET orders using quoteOrderQty follow [LOT\_SIZE](https://developers.binance.com/docs/binance-spot-api-docs/testnet/filters.md#lot_size) filter rules.

The order will execute a quantity that has notional value as close as possible to requested quoteOrderQty.

**Data Source:** Matching Engine

**Response:**

Response format is selected by using the newOrderRespType parameter.

ACK response type:

{  
 "id": "56374a46-3061-486b-a311-99ee972eb648",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1, *// always -1 for singular orders*  
 "clientOrderId": "4d96324ff9d44481926157ec08158a40",  
 "transactTime": 1660801715639  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

RESULT response type:

{  
 "id": "56374a46-3061-486b-a311-99ee972eb648",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1, *// always -1 for singular orders*  
 "clientOrderId": "4d96324ff9d44481926157ec08158a40",  
 "transactTime": 1660801715639,  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "workingTime": 1660801715639,  
 "selfTradePreventionMode": "NONE"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000  
,  
 "count": 1  
 }  
 ]  
}

FULL response type:

{  
 "id": "56374a46-3061-486b-a311-99ee972eb648",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "clientOrderId": "4d96324ff9d44481926157ec08158a40",  
 "transactTime": 1660801715793,  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00847000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "198.33521500",  
 "status": "FILLED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "workingTime": 1660801715793,  
 *// FULL response is identical to RESULT response, with the same optional fields*  
 *// based on the order type and parameters. FULL response additionally includes*  
 *// the list of trades which immediately filled the order.*  
 "fills": [  
 {  
 "price": "23416.10000000",  
 "qty": "0.00635000",  
 "commission": "0.000000",  
 "commissionAsset": "BNB",  
 "tradeId": 1650422481  
 },  
 {  
 "price": "23416.50000000",  
 "qty": "0.00212000",  
 "commission": "0.000000",  
 "commissionAsset": "BNB",  
 "tradeId": 1650422482  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Conditional fields in Order Responses**

There are fields in the order responses (e.g. order placement, order query, order cancellation) that appear only if certain conditions are met.

These fields can apply to Order lists.

The fields are listed below:

| Field | Description | Visibility conditions | Examples |
| --- | --- | --- | --- |
| icebergQty | Quantity for the iceberg order | Appears only if the parameter icebergQty was sent in the request. | "icebergQty": "0.00000000" |
| preventedMatchId | When used in combination with symbol, can be used to query a prevented match. | Appears only if the order expired due to STP. | "preventedMatchId": 0 |
| preventedQuantity | Order quantity that expired due to STP | Appears only if the order expired due to STP. | "preventedQuantity": "1.200000" |
| stopPrice | Price when the algorithmic order will be triggered | Appears for STOP\_LOSS. TAKE\_PROFIT, STOP\_LOSS\_LIMIT and TAKE\_PROFIT\_LIMIT orders. | "stopPrice": "23500.00000000" |
| strategyId | Can be used to label an order that's part of an order strategy. | Appears if the parameter was populated in the request. | "strategyId": 37463720 |
| strategyType | Can be used to label an order that is using an order strategy. | Appears if the parameter was populated in the request. | "strategyType": 1000000 |
| trailingDelta | Delta price change required before order activation | Appears for Trailing Stop Orders. | "trailingDelta": 10 |
| trailingTime | Time when the trailing order is now active and tracking price changes | Appears only for Trailing Stop Orders. | "trailingTime": -1 |
| usedSor | Field that determines whether order used SOR | Appears when placing orders using SOR | "usedSor": true |
| workingFloor | Field that determines whether the order is being filled by the SOR or by the order book the order was submitted to. | Appears when placing orders using SOR | "workingFloor": "SOR" |

**Test new order (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#test-new-order-trade)

{  
 "id": "6ffebe91-01d9-43ac-be99-57cf062e0e30",  
 "method": "order.test",  
 "params": {  
 "symbol": "BTCUSDT",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "price": "23416.10000000",  
 "quantity": "0.00847000",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "15af09e41c36f3cc61378c2fbe2c33719a03dd5eba8d0f9206fbda44de717c88",  
 "timestamp": 1660801715431  
 }  
}

Test order placement.

Validates new order parameters and verifies your signature but does not send the order into the matching engine.

**Weight:**

| Condition | Request Weight |
| --- | --- |
| Without computeCommissionRates | 1 |
| With computeCommissionRates | 20 |

**Parameters:**

In addition to all parameters accepted by [order.place](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-trade), the following optional parameters are also accepted:

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| computeCommissionRates | BOOLEAN | NO | Default: false |

**Data Source:** Memory

**Response:**

Without computeCommissionRates:

{  
 "id": "6ffebe91-01d9-43ac-be99-57cf062e0e30",  
 "status": 200,  
 "result": {},  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

With computeCommissionRates:

{  
 "id": "6ffebe91-01d9-43ac-be99-57cf062e0e30",  
 "status": 200,  
 "result": {  
 "standardCommissionForOrder": { *//Standard commission rates on trades from the order.*  
 "maker": "0.00000112",  
 "taker": "0.00000114"  
 },  
 "taxCommissionForOrder": { *//Tax commission rates for trades from the order*  
 "maker": "0.00000112",  
 "taker": "0.00000114"  
 },  
 "discount": { *//Discount on standard commissions when paying in BNB.*  
 "enabledForAccount": true,  
 "enabledForSymbol": true,  
 "discountAsset": "BNB",  
 "discount": "0.25000000" *//Standard commission is reduced by this rate when paying in BNB.*  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Cancel order (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-order-trade)

{  
 "id": "5633b6a2-90a9-4192-83e7-925c90b6a2fd",  
 "method": "order.cancel",  
 "params": {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "4d96324ff9d44481926157",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "33d5b721f278ae17a52f004a82a6f68a70c68e7dd6776ed0be77a455ab855282",  
 "timestamp": 1660801715830  
 }  
}

Cancel an active order.

**Weight:** 1

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderId | INT | YES | Cancel order by orderId |
| origClientOrderId | STRING | Cancel order by clientOrderId |
| newClientOrderId | STRING | NO | New ID for the canceled order. Automatically generated if not sent |
| cancelRestrictions | ENUM | NO | Supported values: ONLY\_NEW - Cancel will succeed if the order status is NEW. ONLY\_PARTIALLY\_FILLED - Cancel will succeed if order status is PARTIALLY\_FILLED. |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* If both orderId and origClientOrderId parameters are provided, the orderId is searched first, then the origClientOrderId from that result is checked against that order. If both conditions are not met the request will be rejected.
* newClientOrderId will replace clientOrderId of the canceled order, freeing it up for new orders.
* If you cancel an order that is a part of an order list, the entire order list is canceled.
* The performance for canceling an order (single cancel or as part of a cancel-replace) is always better when only orderId is sent. Sending origClientOrderId or both orderId + origClientOrderId will be slower.

**Data Source:** Matching Engine

**Response:**

When an individual order is canceled:

{  
 "id": "5633b6a2-90a9-4192-83e7-925c90b6a2fd",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "4d96324ff9d44481926157", *// clientOrderId that was canceled*  
 "orderId": 12569099453,  
 "orderListId": -1, *// set only for legs of an order list*  
 "clientOrderId": "91fe37ce9e69c90d6358c0", *// newClientOrderId from request*  
 "transactTime": 1684804350068,  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00001000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.23416100",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "stopPrice": "0.00000000", *// present only if stopPrice set for the order*  
 "trailingDelta": 0, *// present only if trailingDelta set for the order*  
 "icebergQty": "0.00000000", *// present only if icebergQty set for the order*  
 "strategyId": 37463720, *// present only if strategyId set for the order*  
 "strategyType": 1000000, *// present only if strategyType set for the order*  
 "selfTradePreventionMode": "NONE"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

When an order list is canceled:

{  
 "id": "16eaf097-bbec-44b9-96ff-e97e6e875870",  
 "status": 200,  
 "result": {  
 "orderListId": 19431,  
 "contingencyType": "OCO",  
 "listStatusType": "ALL\_DONE",  
 "listOrderStatus": "ALL\_DONE",  
 "listClientOrderId": "iuVNVJYYrByz6C4yGOPPK0",  
 "transactionTime": 1660803702431,  
 "symbol": "BTCUSDT",  
 "orders": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "clientOrderId": "bX5wROblo6YeDwa9iTLeyY"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099454,  
 "clientOrderId": "Tnu2IP0J5Y4mxw3IATBfmW"  
 }  
 ],  
 *//order list order's status format is the same as for individual orders.*  
 "orderReports": [  
 {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "bX5wROblo6YeDwa9iTLeyY",  
 "orderId": 12569099453,  
 "orderListId": 19431,  
 "clientOrderId": "OFFXQtxVFZ6Nbcg4PgE2DA",  
 "transactTime": 1684804350068,  
 "price": "23450.50000000",  
 "origQty": "0.00850000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "STOP\_LOSS\_LIMIT",  
 "side": "BUY",  
 "stopPrice": "23430.00000000",  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "Tnu2IP0J5Y4mxw3IATBfmW",  
 "orderId": 12569099454,  
 "orderListId": 19431,  
 "clientOrderId": "OFFXQtxVFZ6Nbcg4PgE2DA",  
 "transactTime": 1684804350068,  
 "price": "23400.00000000",  
 "origQty": "0.00850000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT\_MAKER",  
 "side": "BUY",  
 "selfTradePreventionMode": "NONE"  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Regarding cancelRestrictions**

* If the cancelRestrictions value is not any of the supported values, the error will be:

{  
 "code": -1145,  
 "msg": "Invalid cancelRestrictions"  
}

* If the order did not pass the conditions for cancelRestrictions, the error will be:

{  
 "code": -2011,  
 "msg": "Order was not canceled due to cancel restrictions."  
}

**Cancel and replace order (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-and-replace-order-trade)

{  
 "id": "99de1036-b5e2-4e0f-9b5c-13d751c93a1a",  
 "method": "order.cancelReplace",  
 "params": {  
 "symbol": "BTCUSDT",  
 "cancelReplaceMode": "ALLOW\_FAILURE",  
 "cancelOrigClientOrderId": "4d96324ff9d44481926157",  
 "side": "SELL",  
 "type": "LIMIT",  
 "timeInForce": "GTC",  
 "price": "23416.10000000",  
 "quantity": "0.00847000",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "7028fdc187868754d25e42c37ccfa5ba2bab1d180ad55d4c3a7e2de643943dc5",  
 "timestamp": 1660813156900  
 }  
}

Cancel an existing order and immediately place a new order instead of the canceled one.

A new order that was not attempted (i.e. when newOrderResult: NOT\_ATTEMPTED), will still increase the unfilled order count by 1.

**Weight:** 1

**Unfilled Order Count:** 1

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| cancelReplaceMode | ENUM | YES |  |
| cancelOrderId | INT | YES | Cancel order by orderId |
| cancelOrigClientOrderId | STRING | Cancel order by clientOrderId |
| cancelNewClientOrderId | STRING | NO | New ID for the canceled order. Automatically generated if not sent |
| side | ENUM | YES | BUY or SELL |
| type | ENUM | YES |  |
| timeInForce | ENUM | NO \* |  |
| price | DECIMAL | NO \* |  |
| quantity | DECIMAL | NO \* |  |
| quoteOrderQty | DECIMAL | NO \* |  |
| newClientOrderId | STRING | NO | Arbitrary unique ID among open orders. Automatically generated if not sent |
| newOrderRespType | ENUM | NO | Select response format: ACK, RESULT, FULL.  MARKET and LIMIT orders produce FULL response by default, other order types default to ACK. |
| stopPrice | DECIMAL | NO \* |  |
| trailingDelta | DECIMAL | NO \* | See [Trailing Stop order FAQ](https://developers.binance.com/docs/binance-spot-api-docs/faqs/trailing-stop-faq) |
| icebergQty | DECIMAL | NO |  |
| strategyId | LONG | NO | Arbitrary numeric value identifying the order within an order strategy. |
| strategyType | INT | NO | Arbitrary numeric value identifying the order strategy.  Values smaller than 1000000 are reserved and cannot be used. |
| selfTradePreventionMode | ENUM | NO | The allowed enums is dependent on what is configured on the symbol.  Supported values: [STP Modes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums.md#stpmodes). |
| cancelRestrictions | ENUM | NO | Supported values: ONLY\_NEW - Cancel will succeed if the order status is NEW. ONLY\_PARTIALLY\_FILLED - Cancel will succeed if order status is PARTIALLY\_FILLED. For more information please refer to [Regarding cancelRestrictions](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#regarding-cancelrestrictions). |
| apiKey | STRING | YES |  |
| orderRateLimitExceededMode | ENUM | NO | Supported values: DO\_NOTHING (default)- will only attempt to cancel the order if account has not exceeded the unfilled order rate limit CANCEL\_ONLY - will always cancel the order. |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Similar to the [order.place](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-trade) request, additional mandatory parameters (\*) are determined by the new order [type](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#order-type).

Available cancelReplaceMode options:

* STOP\_ON\_FAILURE – if cancellation request fails, new order placement will not be attempted.
* ALLOW\_FAILURE – new order placement will be attempted even if the cancel request fails.

| Request | | | Response | | |
| --- | --- | --- | --- | --- | --- |
| cancelReplaceMode | orderRateLimitExceededMode | Unfilled Order Count | cancelResult | newOrderResult | status |
| STOP\_ON\_FAILURE | DO\_NOTHING | Within Limits | ✅ SUCCESS | ✅ SUCCESS | 200 |
| ❌ FAILURE | ➖ NOT\_ATTEMPTED | 400 |
| ✅ SUCCESS | ❌ FAILURE | 409 |
| Exceeds Limits | ✅ SUCCESS | ✅ SUCCESS | N/A |
| ❌ FAILURE | ➖ NOT\_ATTEMPTED | N/A |
| ✅ SUCCESS | ❌ FAILURE | N/A |
| CANCEL\_ONLY | Within Limits | ✅ SUCCESS | ✅ SUCCESS | 200 |
| ❌ FAILURE | ➖ NOT\_ATTEMPTED | 400 |
| ✅ SUCCESS | ❌ FAILURE | 409 |
| Exceeds Limits | ❌ FAILURE | ➖ NOT\_ATTEMPTED | 429 |
| ✅ SUCCESS | ❌ FAILURE | 429 |
| ALLOW\_FAILURE | DO\_NOTHING | Within Limits | ✅ SUCCESS | ✅ SUCCESS | 200 |
| ❌ FAILURE | ❌ FAILURE | 400 |
| ❌ FAILURE | ✅ SUCCESS | 409 |
| ✅ SUCCESS | ❌ FAILURE | 409 |
| Exceeds Limits | ✅ SUCCESS | ✅ SUCCESS | N/A |
| ❌ FAILURE | ❌ FAILURE | N/A |
| ❌ FAILURE | ✅ SUCCESS | N/A |
| ✅ SUCCESS | ❌ FAILURE | N/A |
| CANCEL\_ONLY | Within Limits | ✅ SUCCESS | ✅ SUCCESS | 200 |
| ❌ FAILURE | ❌ FAILURE | 400 |
| ❌ FAILURE | ✅ SUCCESS | 409 |
| ✅ SUCCESS | ❌ FAILURE | 409 |
| Exceeds Limits | ✅ SUCCESS | ✅ SUCCESS | 200 |
| ❌ FAILURE | ❌ FAILURE | 400 |
| ❌ FAILURE | ✅ SUCCESS | N/A |
| ✅ SUCCESS | ❌ FAILURE | 409 |

Notes:

* If both cancelOrderId and cancelOrigClientOrderId parameters are provided, the cancelOrderId is searched first, then the cancelOrigClientOrderId from that result is checked against that order. If both conditions are not met the request will be rejected.
* cancelNewClientOrderId will replace clientOrderId of the canceled order, freeing it up for new orders.
* newClientOrderId specifies clientOrderId value for the placed order.

A new order with the same clientOrderId is accepted only when the previous one is filled or expired.

The new order can reuse old clientOrderId of the canceled order.

* This cancel-replace operation is **not transactional**.

If one operation succeeds but the other one fails, the successful operation is still executed.

For example, in STOP\_ON\_FAILURE mode, if the new order placement fails, the old order is still canceled.

* Filters and order count limits are evaluated before cancellation and order placement occurs.
* If new order placement is not attempted, your order count is still incremented.
* Like [order.cancel](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-order-trade), if you cancel an individual order from an order list, the entire order list is canceled.
* The performance for canceling an order (single cancel or as part of a cancel-replace) is always better when only orderId is sent. Sending origClientOrderId or both orderId + origClientOrderId will be slower.

**Data Source:** Matching Engine

**Response:**

If both cancel and placement succeed, you get the following response with "status": 200:

{  
 "id": "99de1036-b5e2-4e0f-9b5c-13d751c93a1a",  
 "status": 200,  
 "result": {  
 "cancelResult": "SUCCESS",  
 "newOrderResult": "SUCCESS",  
 *// Format is identical to "order.cancel" format.*  
 *// Some fields are optional and are included only for orders that set them.*  
 "cancelResponse": {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "4d96324ff9d44481926157", *// cancelOrigClientOrderId from request*  
 "orderId": 125690984230,  
 "orderListId": -1,  
 "clientOrderId": "91fe37ce9e69c90d6358c0", *// cancelNewClientOrderId from request*  
 "transactTime": 1684804350068,  
 "price": "23450.00000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00001000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.23450000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "selfTradePreventionMode": "NONE"  
 },  
 *// Format is identical to "order.place" format, affected by "newOrderRespType".*  
 *// Some fields are optional and are included only for orders that set them.*  
 "newOrderResponse": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "clientOrderId": "bX5wROblo6YeDwa9iTLeyY", *// newClientOrderId from request*  
 "transactTime": 1660813156959,  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "selfTradePreventionMode": "NONE"  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

In STOP\_ON\_FAILURE mode, failed order cancellation prevents new order from being placed and returns the following response with "status": 400:

{  
 "id": "27e1bf9f-0539-4fb0-85c6-06183d36f66c",  
 "status": 400,  
 "error": {  
 "code": -2022,  
 "msg": "Order cancel-replace failed.",  
 "data": {  
 "cancelResult": "FAILURE",  
 "newOrderResult": "NOT\_ATTEMPTED",  
 "cancelResponse": {  
 "code": -2011,  
 "msg": "Unknown order sent."  
 },  
 "newOrderResponse": null  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

If cancel-replace mode allows failure and one of the operations fails, you get a response with "status": 409, and the "data" field detailing which operation succeeded, which failed, and why:

{  
 "id": "b220edfe-f3c4-4a3a-9d13-b35473783a25",  
 "status": 409,  
 "error": {  
 "code": -2021,  
 "msg": "Order cancel-replace partially failed.",  
 "data": {  
 "cancelResult": "SUCCESS",  
 "newOrderResult": "FAILURE",  
 "cancelResponse": {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "4d96324ff9d44481926157",  
 "orderId": 125690984230,  
 "orderListId": -1,  
 "clientOrderId": "91fe37ce9e69c90d6358c0",  
 "price": "23450.00000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00001000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.23450000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "selfTradePreventionMode": "NONE"  
 },  
 "newOrderResponse": {  
 "code": -2010,  
 "msg": "Order would immediately match and take."  
 }  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

{  
 "id": "ce641763-ff74-41ac-b9f7-db7cbe5e93b1",  
 "status": 409,  
 "error": {  
 "code": -2021,  
 "msg": "Order cancel-replace partially failed.",  
 "data": {  
 "cancelResult": "FAILURE",  
 "newOrderResult": "SUCCESS",  
 "cancelResponse": {  
 "code": -2011,  
 "msg": "Unknown order sent."  
 },  
 "newOrderResponse": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "clientOrderId": "bX5wROblo6YeDwa9iTLeyY",  
 "transactTime": 1660813156959,  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "workingTime": 1669693344508,  
 "fills": [],  
 "selfTradePreventionMode": "NONE"  
 }  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

If both operations fail, response will have "status": 400:

{  
 "id": "3b3ac45c-1002-4c7d-88e8-630c408ecd87",  
 "status": 400,  
 "error": {  
 "code": -2022,  
 "msg": "Order cancel-replace failed.",  
 "data": {  
 "cancelResult": "FAILURE",  
 "newOrderResult": "FAILURE",  
 "cancelResponse": {  
 "code": -2011,  
 "msg": "Unknown order sent."  
 },  
 "newOrderResponse": {  
 "code": -2010,  
 "msg": "Order would immediately match and take."  
 }  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 1  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 1  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

If orderRateLimitExceededMode is DO\_NOTHING regardless of cancelReplaceMode, and you have exceeded your unfilled order count, you will get status 429 with the following error:

{  
 "id": "3b3ac45c-1002-4c7d-88e8-630c408ecd87",  
 "status": 429,  
 "error": {  
 "code": -1015,  
 "msg": "Too many new orders; current limit is 50 orders per 10 SECOND."  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 50  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 50  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

If orderRateLimitExceededMode is CANCEL\_ONLY regardless of cancelReplaceMode, and you have exceeded your unfilled order count, you will get status 409 with the following error:

{  
 "id": "3b3ac45c-1002-4c7d-88e8-630c408ecd87",  
 "status": 409,  
 "error": {  
 "code": -2021,  
 "msg": "Order cancel-replace partially failed.",  
 "data": {  
 "cancelResult": "SUCCESS",  
 "newOrderResult": "FAILURE",  
 "cancelResponse": {  
 "symbol": "LTCBNB",  
 "origClientOrderId": "GKt5zzfOxRDSQLveDYCTkc",  
 "orderId": 64,  
 "orderListId": -1,  
 "clientOrderId": "loehOJF3FjoreUBDmv739R",  
 "transactTime": 1715779007228,  
 "price": "1.00",  
 "origQty": "10.00000000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "selfTradePreventionMode": "NONE"  
 },  
 "newOrderResponse": {  
 "code": -1015,  
 "msg": "Too many new orders; current limit is 50 orders per 10 SECOND."  
 }  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 50  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 50  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Order Amend Keep Priority (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#order-amend-keep-priority-trade)

{  
 "id": "56374a46-3061-486b-a311-89ee972eb648",  
 "method": "order.amend.keepPriority",  
 "params": {  
 "newQty": "5",  
 "origClientOrderId": "my\_test\_order1",  
 "recvWindow": 5000,  
 "symbol": "BTCUSDT",  
 "timestamp": 1741922620419,  
 "apiKey": "Rl1KOMDCpSg6xviMYOkNk9ENUB5QOTnufXukVe0Ijd40yduAlpHn78at3rJyJN4F",  
 "signature": "fa49c0c4ebc331c6ebd3fcb20deb387f60081ea858eebe6e35aa6fcdf2a82e08"  
 }  
}

Reduce the quantity of an existing open order.

This adds 0 orders to the EXCHANGE\_MAX\_ORDERS filter and the MAX\_NUM\_ORDERS filter.

Read [Order Amend Keep Priority FAQ](https://developers.binance.com/docs/binance-spot-api-docs/faqs/order_amend_keep_priority) to learn more.

**Weight**: 4

**Unfilled Order Count:** 0

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderId | LONG | NO\* | orderId or origClientOrderId must be sent |
| origClientOrderId | STRING | NO\* | orderId or origClientOrderId must be sent |
| newClientOrderId | STRING | NO\* | The new client order ID for the order after being amended. If not sent, one will be randomly generated. It is possible to reuse the current clientOrderId by sending it as the newClientOrderId. |
| newQty | DECIMAL | YES | newQty must be greater than 0 and less than the order's quantity. |
| recvWindow | LONG | NO | The value cannot be greater than 60000. |
| timestamp | LONG | YES |  |

**Data Source**: Matching Engine

**Response:**

Response for a single order:

{  
 "id": "56374a46-3061-486b-a311-89ee972eb648",  
 "status": 200,  
 "result":  
 {  
 "transactTime": 1741923284382,  
 "executionId": 16,  
 "amendedOrder":  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12,  
 "orderListId": -1,  
 "origClientOrderId": "my\_test\_order1",  
 "clientOrderId": "4zR9HFcEq8gM1tWUqPEUHc",  
 "price": "5.00000000",  
 "qty": "5.00000000",  
 "executedQty": "0.00000000",  
 "preventedQty": "0.00000000",  
 "quoteOrderQty": "0.00000000",  
 "cumulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "BUY",  
 "workingTime": 1741923284364,  
 "selfTradePreventionMode": "NONE"  
 }  
 },  
 "rateLimits":  
 [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

Response for an order which is part of an Order list:

{  
 "id": "56374b46-3061-486b-a311-89ee972eb648",  
 "status": 200,  
 "result":  
 {  
 "transactTime": 1741924229819,  
 "executionId": 60,  
 "amendedOrder":  
 {  
 "symbol": "BTUCSDT",  
 "orderId": 23,  
 "orderListId": 4,  
 "origClientOrderId": "my\_pending\_order",  
 "clientOrderId": "xbxXh5SSwaHS7oUEOCI88B",  
 "price": "1.00000000",  
 "qty": "5.00000000",  
 "executedQty": "0.00000000",  
 "preventedQty": "0.00000000",  
 "quoteOrderQty": "0.00000000",  
 "cumulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "BUY",  
 "workingTime": 1741924204920,  
 "selfTradePreventionMode": "NONE"  
 },  
 "listStatus":  
 {  
 "orderListId": 4,  
 "contingencyType": "OTO",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "8nOGLLawudj1QoOiwbroRH",  
 "symbol": "BTCUSDT",  
 "orders":  
 [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 22,  
 "clientOrderId": "g04EWsjaackzedjC9wRkWD"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 23,  
 "clientOrderId": "xbxXh5SSwaHS7oUEOCI88B"  
 }  
 ]  
 }  
 },  
 "rateLimits":  
 [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Note:** The payloads above do not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Cancel open orders (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-open-orders-trade)

{  
 "id": "778f938f-9041-4b88-9914-efbf64eeacc8",  
 "method": "openOrders.cancelAll",  
 "params": {  
 "symbol": "BTCUSDT",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "773f01b6e3c2c9e0c1d217bc043ce383c1ddd6f0e25f8d6070f2b66a6ceaf3a5",  
 "timestamp": 1660805557200  
 }  
}

Cancel all open orders on a symbol. This includes orders that are part of an order list.

**Weight:** 1

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

**Data Source:** Matching Engine

**Response:**

Cancellation reports for orders and order lists have the same format as in [order.cancel](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-order-trade).

{  
 "id": "778f938f-9041-4b88-9914-efbf64eeacc8",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "4d96324ff9d44481926157",  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "clientOrderId": "91fe37ce9e69c90d6358c0",  
 "transactTime": 1684804350068,  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00001000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.23416100",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "stopPrice": "0.00000000",  
 "trailingDelta": 0,  
 "trailingTime": -1,  
 "icebergQty": "0.00000000",  
 "strategyId": 37463720,  
 "strategyType": 1000000,  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "orderListId": 19431,  
 "contingencyType": "OCO",  
 "listStatusType": "ALL\_DONE",  
 "listOrderStatus": "ALL\_DONE",  
 "listClientOrderId": "iuVNVJYYrByz6C4yGOPPK0",  
 "transactionTime": 1660803702431,  
 "symbol": "BTCUSDT",  
 "orders": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "clientOrderId": "bX5wROblo6YeDwa9iTLeyY"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099454,  
 "clientOrderId": "Tnu2IP0J5Y4mxw3IATBfmW"  
 }  
 ],  
 "orderReports": [  
 {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "bX5wROblo6YeDwa9iTLeyY",  
 "orderId": 12569099453,  
 "orderListId": 19431,  
 "clientOrderId": "OFFXQtxVFZ6Nbcg4PgE2DA",  
 "transactTime": 1684804350068,  
 "price": "23450.50000000",  
 "origQty": "0.00850000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "STOP\_LOSS\_LIMIT",  
 "side": "BUY",  
 "stopPrice": "23430.00000000",  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "origClientOrderId": "Tnu2IP0J5Y4mxw3IATBfmW",  
 "orderId": 12569099454,  
 "orderListId": 19431,  
 "clientOrderId": "OFFXQtxVFZ6Nbcg4PgE2DA",  
 "transactTime": 1684804350068,  
 "price": "23400.00000000",  
 "origQty": "0.00850000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT\_MAKER",  
 "side": "BUY",  
 "selfTradePreventionMode": "NONE"  
 }  
 ]  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Order lists**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#order-lists)

**Place new Order list - OCO (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-list---oco-trade)

{  
 "id": "56374a46-3261-486b-a211-99ed972eb648",  
 "method": "orderList.place.oco",  
 "params":  
 {  
 "symbol": "LTCBNB",  
 "side": "BUY",  
 "quantity": 1,  
 "timestamp": 1711062760647,  
 "aboveType": "STOP\_LOSS\_LIMIT",  
 "abovePrice": "1.5",  
 "aboveStopPrice": "1.50000001",  
 "aboveTimeInForce": "GTC",  
 "belowType": "LIMIT\_MAKER",  
 "belowPrice": "1.49999999",  
 "apiKey": "duwNf97YPLqhFIk7kZF0dDdGYVAXStA7BeEz0fIT9RAhUbixJtyS6kJ3hhzJsRXC",  
 "signature": "64614cfd8dd38260d4fd86d3c455dbf4b9d1c8a8170ea54f700592a986c30ddb"  
 }  
}

Send in an one-cancels-the-other (OCO) pair, where activation of one order immediately cancels the other.

* An OCO has 2 orders called the **above order** and **below order**.
* One of the orders must be a LIMIT\_MAKER/TAKE\_PROFIT/TAKE\_PROFIT\_LIMIT order and the other must be STOP\_LOSS or STOP\_LOSS\_LIMIT order.
* Price restrictions:
  + If the OCO is on the SELL side:
    - LIMIT\_MAKER/TAKE\_PROFIT\_LIMIT price > Last Traded Price > STOP\_LOSS/STOP\_LOSS\_LIMIT stopPrice
    - TAKE\_PROFIT stopPrice > Last Traded Price > STOP\_LOSS/STOP\_LOSS\_LIMIT stopPrice
  + If the OCO is on the BUY side:
    - LIMIT\_MAKER price < Last Traded Price < STOP\_LOSS/STOP\_LOSS\_LIMIT stopPrice
    - TAKE\_PROFIT stopPrice > Last Traded Price > STOP\_LOSS/STOP\_LOSS\_LIMIT stopPrice
* OCOs add **2 orders** to the EXCHANGE\_MAX\_ORDERS filter and MAX\_NUM\_ORDERS filter.

**Weight:** 1

**Unfilled Order Count:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| listClientOrderId | STRING | NO | Arbitrary unique ID among open order lists. Automatically generated if not sent. A new order list with the same listClientOrderId is accepted only when the previous one is filled or completely expired. listClientOrderId is distinct from the aboveClientOrderId and the belowCLientOrderId. |
| side | ENUM | YES | BUY or SELL |
| quantity | DECIMAL | YES | Quantity for both orders of the order list. |
| aboveType | ENUM | YES | Supported values: STOP\_LOSS\_LIMIT, STOP\_LOSS, LIMIT\_MAKER, TAKE\_PROFIT, TAKE\_PROFIT\_LIMIT |
| aboveClientOrderId | STRING | NO | Arbitrary unique ID among open orders for the above order. Automatically generated if not sent |
| aboveIcebergQty | LONG | NO | Note that this can only be used if aboveTimeInForce is GTC. |
| abovePrice | DECIMAL | NO | Can be used if aboveType is STOP\_LOSS\_LIMIT , LIMIT\_MAKER, or TAKE\_PROFIT\_LIMIT to specify the limit price. |
| aboveStopPrice | DECIMAL | NO | Can be used if aboveType is STOP\_LOSS, STOP\_LOSS\_LIMIT, TAKE\_PROFIT, TAKE\_PROFIT\_LIMIT. Either aboveStopPrice or aboveTrailingDelta or both, must be specified. |
| aboveTrailingDelta | LONG | NO | See [Trailing Stop order FAQ](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/..faqs/trailing-stop-faq.md). |
| aboveTimeInForce | DECIMAL | NO | Required if aboveType is STOP\_LOSS\_LIMIT or TAKE\_PROFIT\_LIMIT. |
| aboveStrategyId | LONG | NO | Arbitrary numeric value identifying the above order within an order strategy. |
| aboveStrategyType | INT | NO | Arbitrary numeric value identifying the above order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| belowType | ENUM | YES | Supported values: STOP\_LOSS, STOP\_LOSS\_LIMIT, TAKE\_PROFIT,TAKE\_PROFIT\_LIMIT |
| belowClientOrderId | STRING | NO |  |
| belowIcebergQty | LONG | NO | Note that this can only be used if belowTimeInForce is GTC. |
| belowPrice | DECIMAL | NO | Can be used if belowType is STOP\_LOSS\_LIMIT , LIMIT\_MAKER, or TAKE\_PROFIT\_LIMIT to specify the limit price. |
| belowStopPrice | DECIMAL | NO | Can be used if belowType is STOP\_LOSS, STOP\_LOSS\_LIMIT, TAKE\_PROFIT or TAKE\_PROFIT\_LIMIT. Either belowStopPrice or belowTrailingDelta or both, must be specified. |
| belowTrailingDelta | LONG | NO | See [Trailing Stop order FAQ](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/..faqs/trailing-stop-faq.md). |
| belowTimeInForce | ENUM | NO | Required if belowType is STOP\_LOSS\_LIMIT or TAKE\_PROFIT\_LIMIT |
| belowStrategyId | LONG | NO | Arbitrary numeric value identifying the below order within an order strategy. |
| belowStrategyType | INT | NO | Arbitrary numeric value identifying the below order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| newOrderRespType | ENUM | NO | Select response format: ACK, RESULT, FULL |
| selfTradePreventionMode | ENUM | NO | The allowed enums is dependent on what is configured on the symbol. The possible supported values are: [STP Modes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#stpmodes). |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000. |
| timestamp | LONG | YES |  |
| signature | STRING | YES |  |

**Data Source:** Matching Engine

**Response:**

Response format for orderReports is selected using the newOrderRespType parameter. The following example is for RESULT response type. See [order.place](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-trade) for more examples.

{  
 "id": "56374a46-3261-486b-a211-99ed972eb648",  
 "status": 200,  
 "result":  
 {  
 "orderListId": 2,  
 "contingencyType": "OCO",  
 "listStatusType": "EXEC\_STARTED",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "cKPMnDCbcLQILtDYM4f4fX",  
 "transactionTime": 1711062760648,  
 "symbol": "LTCBNB",  
 "orders":  
 [  
 {  
 "symbol": "LTCBNB",  
 "orderId": 2,  
 "clientOrderId": "0m6I4wfxvTUrOBSMUl0OPU"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 3,  
 "clientOrderId": "Z2IMlR79XNY5LU0tOxrWyW"  
 }  
 ],  
 "orderReports":  
 [  
 {  
 "symbol": "LTCBNB",  
 "orderId": 2,  
 "orderListId": 2,  
 "clientOrderId": "0m6I4wfxvTUrOBSMUl0OPU",  
 "transactTime": 1711062760648,  
 "price": "1.50000000",  
 "origQty": "1.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "STOP\_LOSS\_LIMIT",  
 "side": "BUY",  
 "stopPrice": "1.50000001",  
 "workingTime": -1,  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 3,  
 "orderListId": 2,  
 "clientOrderId": "Z2IMlR79XNY5LU0tOxrWyW",  
 "transactTime": 1711062760648,  
 "price": "1.49999999",  
 "origQty": "1.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT\_MAKER",  
 "side": "BUY",  
 "workingTime": 1711062760648,  
 "selfTradePreventionMode": "NONE"  
 }  
 ]  
 },  
 "rateLimits":  
 [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 2  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 2  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Place new Order list - OTO (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-list---oto-trade)

{  
 "id": "1712544395950",  
 "method": "orderList.place.oto",  
 "params": {  
 "signature": "3e1e5ac8690b0caf9a2afd5c5de881ceba69939cc9d817daead5386bf65d0cbb",  
 "apiKey": "Rf07JlnL9PHVxjs27O5CvKNyOsV4qJ5gXdrRfpvlOdvMZbGZbPO5Ce2nIwfRP0iA",  
 "pendingQuantity": 1,  
 "pendingSide": "BUY",  
 "pendingType": "MARKET",  
 "symbol": "LTCBNB",  
 "recvWindow": "5000",  
 "timestamp": "1712544395951",  
 "workingPrice": 1,  
 "workingQuantity": 1,  
 "workingSide": "SELL",  
 "workingTimeInForce": "GTC",  
 "workingType": "LIMIT"  
 }  
}

Places an OTO.

* An OTO (One-Triggers-the-Other) is an order list comprised of 2 orders.
* The first order is called the **working order** and must be LIMIT or LIMIT\_MAKER. Initially, only the working order goes on the order book.
* The second order is called the **pending order**. It can be any order type except for MARKET orders using parameter quoteOrderQty. The pending order is only placed on the order book when the working order gets **fully filled**.
* If either the working order or the pending order is cancelled individually, the other order in the order list will also be canceled or expired.
* When the order list is placed, if the working order gets **immediately fully filled**, the placement response will show the working order as FILLED but the pending order will still appear as PENDING\_NEW. You need to query the status of the pending order again to see its updated status.
* OTOs add **2 orders** to the EXCHANGE\_MAX\_NUM\_ORDERS filter and MAX\_NUM\_ORDERS filter.

**Weight:** 1

**Unfilled Order Count:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| listClientOrderId | STRING | NO | Arbitrary unique ID among open order lists. Automatically generated if not sent. A new order list with the same listClientOrderId is accepted only when the previous one is filled or completely expired. listClientOrderId is distinct from the workingClientOrderId and the pendingClientOrderId. |
| newOrderRespType | ENUM | NO | Format of the JSON response. Supported values: [Order Response Type](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#orderresponsetype) |
| selfTradePreventionMode | ENUM | NO | The allowed values are dependent on what is configured on the symbol. Supported values: [STP Modes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#stpmodes) |
| workingType | ENUM | YES | Supported values: LIMIT,LIMIT\_MAKER |
| workingSide | ENUM | YES | Supported values: [Order side](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#side) |
| workingClientOrderId | STRING | NO | Arbitrary unique ID among open orders for the working order. Automatically generated if not sent. |
| workingPrice | DECIMAL | YES |  |
| workingQuantity | DECIMAL | YES | Sets the quantity for the working order. |
| workingIcebergQty | DECIMAL | NO | This can only be used if workingTimeInForce is GTC, or if workingType is LIMIT\_MAKER. |
| workingTimeInForce | ENUM | NO | Supported values: [Time In Force](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#timeinforce) |
| workingStrategyId | LONG | NO | Arbitrary numeric value identifying the working order within an order strategy. |
| workingStrategyType | INT | NO | Arbitrary numeric value identifying the working order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| pendingType | ENUM | YES | Supported values: [Order types](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#order-type). Note that MARKET orders using quoteOrderQty are not supported. |
| pendingSide | ENUM | YES | Supported values: [Order side](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#side) |
| pendingClientOrderId | STRING | NO | Arbitrary unique ID among open orders for the pending order. Automatically generated if not sent. |
| pendingPrice | DECIMAL | NO |  |
| pendingStopPrice | DECIMAL | NO |  |
| pendingTrailingDelta | DECIMAL | NO |  |
| pendingQuantity | DECIMAL | YES | Sets the quantity for the pending order. |
| pendingIcebergQty | DECIMAL | NO | This can only be used if pendingTimeInForce is GTC, or if pendingType is LIMIT\_MAKER. |
| pendingTimeInForce | ENUM | NO | Supported values: [Time In Force](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#timeinforce) |
| pendingStrategyId | LONG | NO | Arbitrary numeric value identifying the pending order within an order strategy. |
| pendingStrategyType | INT | NO | Arbitrary numeric value identifying the pending order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| recvWindow | LONG | NO | The value cannot be greater than 60000. |
| timestamp | LONG | YES |  |
| signature | STRING | YES |  |

**Mandatory parameters based on pendingType or workingType**

Depending on the pendingType or workingType, some optional parameters will become mandatory.

| Type | Additional mandatory parameters | Additional information |
| --- | --- | --- |
| workingType = LIMIT | workingTimeInForce |  |
| pendingType = LIMIT | pendingPrice, pendingTimeInForce |  |
| pendingType = STOP\_LOSS or TAKE\_PROFIT | pendingStopPrice and/or pendingTrailingDelta |  |
| pendingType =STOP\_LOSS\_LIMIT or TAKE\_PROFIT\_LIMIT | pendingPrice, pendingStopPrice and/or pendingTrailingDelta, pendingTimeInForce |  |

**Data Source:**

Matching Engine

**Response:**

{  
 "id": "1712544395950",  
 "status": 200,  
 "result": {  
 "orderListId": 626,  
 "contingencyType": "OTO",  
 "listStatusType": "EXEC\_STARTED",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "KA4EBjGnzvSwSCQsDdTrlf",  
 "transactionTime": 1712544395981,  
 "symbol": "1712544378871",  
 "orders": [  
 {  
 "symbol": "LTCBNB",  
 "orderId": 13,  
 "clientOrderId": "YiAUtM9yJjl1a2jXHSp9Ny"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 14,  
 "clientOrderId": "9MxJSE1TYkmyx5lbGLve7R"  
 }  
 ],  
 "orderReports": [  
 {  
 "symbol": "LTCBNB",  
 "orderId": 13,  
 "orderListId": 626,  
 "clientOrderId": "YiAUtM9yJjl1a2jXHSp9Ny",  
 "transactTime": 1712544395981,  
 "price": "1.000000",  
 "origQty": "1.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "workingTime": 1712544395981,  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 14,  
 "orderListId": 626,  
 "clientOrderId": "9MxJSE1TYkmyx5lbGLve7R",  
 "transactTime": 1712544395981,  
 "price": "0.000000",  
 "origQty": "1.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.000000",  
 "status": "PENDING\_NEW",  
 "timeInForce": "GTC",  
 "type": "MARKET",  
 "side": "BUY",  
 "workingTime": -1,  
 "selfTradePreventionMode": "NONE"  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 10000000,  
 "count": 10  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 1000,  
 "count": 38  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Place new Order list - OTOCO (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-list---otoco-trade)

{  
 "id": "1712544408508",  
 "method": "orderList.place.otoco",  
 "params": {  
 "signature": "c094473304374e1b9c5f7e2558358066cfa99df69f50f63d09cfee755136cb07",  
 "apiKey": "Rf07JlnL9PHVxjs27O5CvKNyOsV4qJ5gXdrRfpvlOdvMZbGZbPO5Ce2nIwfRP0iA",  
 "pendingQuantity": 5,  
 "pendingSide": "SELL",  
 "pendingBelowPrice": 5,  
 "pendingBelowType": "LIMIT\_MAKER",  
 "pendingAboveStopPrice": 0.5,  
 "pendingAboveType": "STOP\_LOSS",  
 "symbol": "LTCBNB",  
 "recvWindow": "5000",  
 "timestamp": "1712544408509",  
 "workingPrice": 1.5,  
 "workingQuantity": 1,  
 "workingSide": "BUY",  
 "workingTimeInForce": "GTC",  
 "workingType": "LIMIT"  
 }  
}

Place an OTOCO.

* An OTOCO (One-Triggers-One-Cancels-the-Other) is an order list comprised of 3 orders.
* The first order is called the **working order** and must be LIMIT or LIMIT\_MAKER. Initially, only the working order goes on the order book.
  + The behavior of the working order is the same as the [OTO](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-list---oto-trade).
* OTOCO has 2 pending orders (pending above and pending below), forming an OCO pair. The pending orders are only placed on the order book when the working order gets **fully filled**.
  + The rules of the pending above and pending below follow the same rules as the [Order list OCO](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#new-order-list---oco-trade).
* OTOCOs add **3 orders** to the EXCHANGE\_MAX\_NUM\_ORDERS filter and MAX\_NUM\_ORDERS filter.

**Weight:** 1

**Unfilled Order Count:** 3

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| listClientOrderId | STRING | NO | Arbitrary unique ID among open order lists. Automatically generated if not sent. A new order list with the same listClientOrderId is accepted only when the previous one is filled or completely expired. listClientOrderId is distinct from the workingClientOrderId, pendingAboveClientOrderId, and the pendingBelowClientOrderId. |
| newOrderRespType | ENUM | NO | Format of the JSON response. Supported values: [Order Response Type](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#orderresponsetype) |
| selfTradePreventionMode | ENUM | NO | The allowed values are dependent on what is configured on the symbol. Supported values: [STP Modes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#stpmodes) |
| workingType | ENUM | YES | Supported values: LIMIT, LIMIT\_MAKER |
| workingSide | ENUM | YES | Supported values: [Order Side](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#side) |
| workingClientOrderId | STRING | NO | Arbitrary unique ID among open orders for the working order. Automatically generated if not sent. |
| workingPrice | DECIMAL | YES |  |
| workingQuantity | DECIMAL | YES |  |
| workingIcebergQty | DECIMAL | NO | This can only be used if workingTimeInForce is GTC. |
| workingTimeInForce | ENUM | NO | Supported values: [Time In Force](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#timeinforce) |
| workingStrategyId | LONG | NO | Arbitrary numeric value identifying the working order within an order strategy. |
| workingStrategyType | INT | NO | Arbitrary numeric value identifying the working order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| pendingSide | ENUM | YES | Supported values: [Order Side](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#side) |
| pendingQuantity | DECIMAL | YES |  |
| pendingAboveType | ENUM | YES | Supported values: STOP\_LOSS\_LIMIT, STOP\_LOSS, LIMIT\_MAKER, TAKE\_PROFIT, TAKE\_PROFIT\_LIMIT |
| pendingAboveClientOrderId | STRING | NO | Arbitrary unique ID among open orders for the pending above order. Automatically generated if not sent. |
| pendingAbovePrice | DECIMAL | NO | Can be used if pendingAboveType is STOP\_LOSS\_LIMIT , LIMIT\_MAKER, or TAKE\_PROFIT\_LIMIT to specify the limit price. |
| pendingAboveStopPrice | DECIMAL | NO | Can be used if pendingAboveType is STOP\_LOSS, STOP\_LOSS\_LIMIT, TAKE\_PROFIT, TAKE\_PROFIT\_LIMIT |
| pendingAboveTrailingDelta | DECIMAL | NO | See [Trailing Stop FAQ](https://developers.binance.com/docs/binance-spot-api-docs/faqs/trailing-stop-faq) |
| pendingAboveIcebergQty | DECIMAL | NO | This can only be used if pendingAboveTimeInForce is GTC or if pendingAboveType is LIMIT\_MAKER. |
| pendingAboveTimeInForce | ENUM | NO |  |
| pendingAboveStrategyId | LONG | NO | Arbitrary numeric value identifying the pending above order within an order strategy. |
| pendingAboveStrategyType | INT | NO | Arbitrary numeric value identifying the pending above order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| pendingBelowType | ENUM | NO | Supported values: STOP\_LOSS, STOP\_LOSS\_LIMIT, TAKE\_PROFIT,TAKE\_PROFIT\_LIMIT |
| pendingBelowClientOrderId | STRING | NO | Arbitrary unique ID among open orders for the pending below order. Automatically generated if not sent. |
| pendingBelowPrice | DECIMAL | NO | Can be used if pendingBelowType is STOP\_LOSS\_LIMIT or TAKE\_PROFIT\_LIMIT to specify the limit price. |
| pendingBelowStopPrice | DECIMAL | NO | Can be used if pendingBelowType is STOP\_LOSS, STOP\_LOSS\_LIMIT, TAKE\_PROFIT or TAKE\_PROFIT\_LIMIT. Either pendingBelowStopPrice or pendingBelowTrailingDelta or both, must be specified. |
| pendingBelowTrailingDelta | DECIMAL | NO |  |
| pendingBelowIcebergQty | DECIMAL | NO | This can only be used if pendingBelowTimeInForce is GTC, or if pendingBelowType is LIMIT\_MAKER. |
| pendingBelowTimeInForce | ENUM | NO | Supported values: [Time In Force](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#timeinforce) |
| pendingBelowStrategyId | LONG | NO | Arbitrary numeric value identifying the pending below order within an order strategy. |
| pendingBelowStrategyType | INT | NO | Arbitrary numeric value identifying the pending below order strategy. Values smaller than 1000000 are reserved and cannot be used. |
| recvWindow | LONG | NO | The value cannot be greater than 60000. |
| timestamp | LONG | YES |  |
| signature | STRING | YES |  |

**Mandatory parameters based on pendingAboveType, pendingBelowType or workingType**

Depending on the pendingAboveType/pendingBelowType or workingType, some optional parameters will become mandatory.

| Type | Additional mandatory parameters | Additional information |
| --- | --- | --- |
| workingType = LIMIT | workingTimeInForce |  |
| pendingAboveType= LIMIT\_MAKER | pendingAbovePrice |  |
| pendingAboveType = STOP\_LOSS/TAKE\_PROFIT | pendingAboveStopPrice and/or pendingAboveTrailingDelta |  |
| pendingAboveType=STOP\_LOSS\_LIMIT/TAKE\_PROFIT\_LIMIT | pendingAbovePrice, pendingAboveStopPrice and/or pendingAboveTrailingDelta, pendingAboveTimeInForce |  |
| pendingBelowType= LIMIT\_MAKER | pendingBelowPrice |  |
| pendingBelowType= STOP\_LOSS/TAKE\_PROFIT | pendingBelowStopPrice and/or pendingBelowTrailingDelta |  |
| pendingBelowType=STOP\_LOSS\_LIMIT/TAKE\_PROFIT\_LIMIT | pendingBelowPrice, pendingBelowStopPrice and/or pendingBelowTrailingDelta, pendingBelowTimeInForce |  |

**Data Source:** Matching Engine

**Response:**

{  
 "id": "1712544408508",  
 "status": 200,  
 "result": {  
 "orderListId": 629,  
 "contingencyType": "OTO",  
 "listStatusType": "EXEC\_STARTED",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "GaeJHjZPasPItFj4x7Mqm6",  
 "transactionTime": 1712544408537,  
 "symbol": "1712544378871",  
 "orders": [  
 {  
 "symbol": "LTCBNB",  
 "orderId": 23,  
 "clientOrderId": "OVQOpKwfmPCfaBTD0n7e7H"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 24,  
 "clientOrderId": "YcCPKCDMQIjNvLtNswt82X"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 25,  
 "clientOrderId": "ilpIoShcFZ1ZGgSASKxMPt"  
 }  
 ],  
 "orderReports": [  
 {  
 "symbol": "LTCBNB",  
 "orderId": 23,  
 "orderListId": 629,  
 "clientOrderId": "OVQOpKwfmPCfaBTD0n7e7H",  
 "transactTime": 1712544408537,  
 "price": "1.500000",  
 "origQty": "1.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.000000",  
 "status": "NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "BUY",  
 "workingTime": 1712544408537,  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 24,  
 "orderListId": 629,  
 "clientOrderId": "YcCPKCDMQIjNvLtNswt82X",  
 "transactTime": 1712544408537,  
 "price": "0.000000",  
 "origQty": "5.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.000000",  
 "status": "PENDING\_NEW",  
 "timeInForce": "GTC",  
 "type": "STOP\_LOSS",  
 "side": "SELL",  
 "stopPrice": "0.500000",  
 "workingTime": -1,  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "LTCBNB",  
 "orderId": 25,  
 "orderListId": 629,  
 "clientOrderId": "ilpIoShcFZ1ZGgSASKxMPt",  
 "transactTime": 1712544408537,  
 "price": "5.000000",  
 "origQty": "5.000000",  
 "executedQty": "0.000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.000000",  
 "status": "PENDING\_NEW",  
 "timeInForce": "GTC",  
 "type": "LIMIT\_MAKER",  
 "side": "SELL",  
 "workingTime": -1,  
 "selfTradePreventionMode": "NONE"  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 10000000,  
 "count": 18  
 },  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 1000,  
 "count": 65  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Cancel Order list (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-order-list-trade)

{  
 "id": "c5899911-d3f4-47ae-8835-97da553d27d0",  
 "method": "orderList.cancel",  
 "params": {  
 "symbol": "BTCUSDT",  
 "orderListId": 1274512,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "4973f4b2fee30bf6d45e4a973e941cc60fdd53c8dd5a25edeac96f5733c0ccee",  
 "timestamp": 1660801720210  
 }  
}

Cancel an active order list.

**Weight**: 1

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderListId | INT | YES | Cancel order list by orderListId |
| listClientOrderId | STRING | Cancel order list by listClientId |
| newClientOrderId | STRING | NO | New ID for the canceled order list. Automatically generated if not sent |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* If both orderListId and listClientOrderId parameters are provided, the orderListId is searched first, then the listClientOrderId from that result is checked against that order. If both conditions are not met the request will be rejected.
* Canceling an individual order with [order.cancel](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#cancel-order-trade) will cancel the entire order list as well.

**Data Source:** Matching Engine

**Response:**

{  
 "id": "c5899911-d3f4-47ae-8835-97da553d27d0",  
 "status": 200,  
 "result": {  
 "orderListId": 1274512,  
 "contingencyType": "OCO",  
 "listStatusType": "ALL\_DONE",  
 "listOrderStatus": "ALL\_DONE",  
 "listClientOrderId": "6023531d7edaad348f5aff",  
 "transactionTime": 1660801720215,  
 "symbol": "BTCUSDT",  
 "orders": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138901,  
 "clientOrderId": "BqtFCj5odMoWtSqGk2X9tU"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138902,  
 "clientOrderId": "jLnZpj5enfMXTuhKB1d0us"  
 }  
 ],  
 "orderReports": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138901,  
 "orderListId": 1274512,  
 "clientOrderId": "BqtFCj5odMoWtSqGk2X9tU",  
 "transactTime": 1660801720215,  
 "price": "23410.00000000",  
 "origQty": "0.00650000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "STOP\_LOSS\_LIMIT",  
 "side": "SELL",  
 "stopPrice": "23405.00000000",  
 "selfTradePreventionMode": "NONE"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138902,  
 "orderListId": 1274512,  
 "clientOrderId": "jLnZpj5enfMXTuhKB1d0us",  
 "transactTime": 1660801720215,  
 "price": "23420.00000000",  
 "origQty": "0.00650000",  
 "executedQty": "0.00000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "0.00000000",  
 "status": "CANCELED",  
 "timeInForce": "GTC",  
 "type": "LIMIT\_MAKER",  
 "side": "SELL",  
 "selfTradePreventionMode": "NONE"  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**SOR**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#sor)

**Place new order using SOR (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-using-sor-trade)

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "method": "sor.order.place",  
 "params":  
 {  
 "symbol": "BTCUSDT",  
 "side": "BUY",  
 "type": "LIMIT",  
 "quantity": 0.5,  
 "timeInForce": "GTC",  
 "price": 31000,  
 "timestamp": 1687485436575,  
 "apiKey": "u5lgqJb97QWXWfgeV4cROuHbReSJM9rgQL0IvYcYc7BVeA5lpAqqc3a5p2OARIFk",  
 "signature": "fd301899567bc9472ce023392160cdc265ad8fcbbb67e0ea1b2af70a4b0cd9c7"  
 }  
}

Places an order using smart order routing (SOR).

This adds 1 order to the EXCHANGE\_MAX\_ORDERS filter and the MAX\_NUM\_ORDERS filter.

Read [SOR FAQ](https://developers.binance.com/docs/binance-spot-api-docs/faqs/sor_faq) to learn more.

**Weight:** 1

**Unfilled Order Count:** 1

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| side | ENUM | YES | BUY or SELL |
| type | ENUM | YES |  |
| timeInForce | ENUM | NO | Applicable only to LIMIT order type |
| price | DECIMAL | NO | Applicable only to LIMIT order type |
| quantity | DECIMAL | YES |  |
| newClientOrderId | STRING | NO | Arbitrary unique ID among open orders. Automatically generated if not sent |
| newOrderRespType | ENUM | NO | Select response format: ACK, RESULT, FULL.  MARKET and LIMIT orders use FULL by default. |
| icebergQty | DECIMAL | NO |  |
| strategyId | LONG | NO | Arbitrary numeric value identifying the order within an order strategy. |
| strategyType | INT | NO | Arbitrary numeric value identifying the order strategy.  Values smaller than 1000000 are reserved and cannot be used. |
| selfTradePreventionMode | ENUM | NO | The allowed enums is dependent on what is configured on the symbol. The possible supported values are: [STP Modes](https://developers.binance.com/docs/binance-spot-api-docs/testnet/enums#stpmodes). |
| apiKey | STRING | YES |  |
| timestamp | LONG | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |

**Note:** sor.order.place only supports LIMIT and MARKET orders. quoteOrderQty is not supported.

**Data Source:** Matching Engine

**Response:**

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 2,  
 "orderListId": -1,  
 "clientOrderId": "sBI1KM6nNtOfj5tccZSKly",  
 "transactTime": 1689149087774,  
 "price": "31000.00000000",  
 "origQty": "0.50000000",  
 "executedQty": "0.50000000",  
 "origQuoteOrderQty": "0.000000",  
 "cummulativeQuoteQty": "14000.00000000",  
 "status": "FILLED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "BUY",  
 "workingTime": 1689149087774,  
 "fills": [  
 {  
 "matchType": "ONE\_PARTY\_TRADE\_REPORT",  
 "price": "28000.00000000",  
 "qty": "0.50000000",  
 "commission": "0.00000000",  
 "commissionAsset": "BTC",  
 "tradeId": -1,  
 "allocId": 0  
 }  
 ],  
 "workingFloor": "SOR",  
 "selfTradePreventionMode": "NONE",  
 "usedSor": true  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

**Test new order using SOR (TRADE)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#test-new-order-using-sor-trade)

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "method": "sor.order.test",  
 "params":  
 {  
 "symbol": "BTCUSDT",  
 "side": "BUY",  
 "type": "LIMIT",  
 "quantity": 0.1,  
 "timeInForce": "GTC",  
 "price": 0.1,  
 "timestamp": 1687485436575,  
 "apiKey": "u5lgqJb97QWXWfgeV4cROuHbReSJM9rgQL0IvYcYc7BVeA5lpAqqc3a5p2OARIFk",  
 "signature": "fd301899567bc9472ce023392160cdc265ad8fcbbb67e0ea1b2af70a4b0cd9c7"  
 }  
}

Test new order creation and signature/recvWindow using smart order routing (SOR). Creates and validates a new order but does not send it into the matching engine.

**Weight:**

| Condition | Request Weight |
| --- | --- |
| Without computeCommissionRates | 1 |
| With computeCommissionRates | 20 |

**Parameters:**

In addition to all parameters accepted by [sor.order.place](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#place-new-order-using-sor-trade), the following optional parameters are also accepted:

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| computeCommissionRates | BOOLEAN | NO | Default: false |

**Data Source:** Memory

**Response:**

Without computeCommissionRates:

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "status": 200,  
 "result": {},  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 1  
 }  
 ]  
}

With computeCommissionRates:

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "status": 200,  
 "result": {  
 "standardCommissionForOrder": { *//Commission rates for the order depending on its role (e.g. maker or taker)*  
 "maker": "0.00000112",  
 "taker": "0.00000114"  
 },  
 "taxCommissionForOrder": { *//Tax deduction rates for the order depending on its role (e.g. maker or taker)*  
 "maker": "0.00000112",  
 "taker": "0.00000114"  
 },  
 "discount": { *//Discount on standard commissions when paying in BNB.*  
 "enabledForAccount": true,  
 "enabledForSymbol": true,  
 "discountAsset": "BNB",  
 "discount": "0.25" *//Standard commission is reduced by this rate when paying in BNB.*  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Account requests**

**Account information (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-information-user_data)

{  
 "id": "605a6d20-6588-4cb9-afa0-b0ab087507ba",  
 "method": "account.status",  
 "params": {  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "83303b4a136ac1371795f465808367242685a9e3a42b22edb4d977d0696eb45c",  
 "timestamp": 1660801839480  
 }  
}

Query information about your account.

**Weight:** 20

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| apiKey | STRING | YES |  |
| omitZeroBalances | BOOLEAN | NO | When set to true, emits only the non-zero balances of an account. Default value: false |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

**Data Source:** Memory => Database

**Response:**

{  
 "id": "605a6d20-6588-4cb9-afa0-b0ab087507ba",  
 "status": 200,  
 "result": {  
 "makerCommission": 15,  
 "takerCommission": 15,  
 "buyerCommission": 0,  
 "sellerCommission": 0,  
 "canTrade": true,  
 "canWithdraw": true,  
 "canDeposit": true,  
 "commissionRates": {  
 "maker": "0.00150000",  
 "taker": "0.00150000",  
 "buyer": "0.00000000",  
 "seller": "0.00000000"  
 },  
 "brokered": false,  
 "requireSelfTradePrevention": false,  
 "preventSor": false,  
 "updateTime": 1660801833000,  
 "accountType": "SPOT",  
 "balances": [  
 {  
 "asset": "BNB",  
 "free": "0.00000000",  
 "locked": "0.00000000"  
 },  
 {  
 "asset": "BTC",  
 "free": "1.3447112",  
 "locked": "0.08600000"  
 },  
 {  
 "asset": "USDT",  
 "free": "1021.21000000",  
 "locked": "0.00000000"  
 }  
 ],  
 "permissions": [  
 "SPOT"  
 ],  
 "uid": 354937868  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000  
,  
 "count": 20  
 }  
 ]  
}

**Query order (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-user_data)

{  
 "id": "aa62318a-5a97-4f3b-bdc7-640bbe33b291",  
 "method": "order.status",  
 "params": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "2c3aab5a078ee4ea465ecd95523b77289f61476c2f238ec10c55ea6cb11a6f35",  
 "timestamp": 1660801720951  
 }  
}

Check execution status of an order.

**Weight:** 4

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderId | INT | YES | Lookup order by orderId |
| origClientOrderId | STRING | Lookup order by clientOrderId |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* If both orderId and origClientOrderId are provided, the orderId is searched first, then the origClientOrderId from that result is checked against that order. If both conditions are not met the request will be rejected.
* For some historical orders the cummulativeQuoteQty response field may be negative, meaning the data is not available at this time.

**Data Source:** Memory => Database

**Response:**

{  
 "id": "aa62318a-5a97-4f3b-bdc7-640bbe33b291",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1, *// set only for orders of an order list*  
 "clientOrderId": "4d96324ff9d44481926157",  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00847000",  
 "cummulativeQuoteQty": "198.33521500",  
 "status": "FILLED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "stopPrice": "0.00000000", *// always present, zero if order type does not use stopPrice*  
 "trailingDelta": 10, *// present only if trailingDelta set for the order*  
 "trailingTime": -1, *// present only if trailingDelta set for the order*  
 "icebergQty": "0.00000000", *// always present, zero for non-iceberg orders*  
 "time": 1660801715639, *// time when the order was placed*  
 "updateTime": 1660801717945, *// time of the last update to the order*  
 "isWorking": true,  
 "workingTime": 1660801715639,  
 "origQuoteOrderQty": "0.00000000" *// always present, zero if order type does not use quoteOrderQty*  
 "strategyId": 37463720, *// present only if strategyId set for the order*  
 "strategyType": 1000000, *// present only if strategyType set for the order*  
 "selfTradePreventionMode": "NONE",  
 "preventedMatchId": 0, *// present only if the order expired due to STP*  
 "preventedQuantity": "1.200000" *// present only if the order expired due to STP*  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Current open orders (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#current-open-orders-user_data)

{  
 "id": "55f07876-4f6f-4c47-87dc-43e5fff3f2e7",  
 "method": "openOrders.status",  
 "params": {  
 "symbol": "BTCUSDT",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "d632b3fdb8a81dd44f82c7c901833309dd714fe508772a89b0a35b0ee0c48b89",  
 "timestamp": 1660813156812  
 }  
}

Query execution status of all open orders.

If you need to continuously monitor order status updates, please consider using WebSocket Streams:

* [userDataStream.start](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#user-data-stream-requests) request
* [executionReport](https://developers.binance.com/docs/binance-spot-api-docs/testnet/user-data-stream#order-update) user data stream event

**Weight:** Adjusted based on the number of requested symbols:

| Parameter | Weight |
| --- | --- |
| symbol | 6 |
| none | 80 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | NO | If omitted, open orders for all symbols are returned |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

**Data Source:** Memory => Database

**Response:**

Status reports for open orders are identical to [order.status](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-user_data).

Note that some fields are optional and included only for orders that set them.

Open orders are always returned as a flat list. If all symbols are requested, use the symbol field to tell which symbol the orders belong to.

{  
 "id": "55f07876-4f6f-4c47-87dc-43e5fff3f2e7",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "clientOrderId": "4d96324ff9d44481926157",  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00720000",  
 "cummulativeQuoteQty": "172.43931000",  
 "status": "PARTIALLY\_FILLED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "stopPrice": "0.00000000",  
 "icebergQty": "0.00000000",  
 "time": 1660801715639,  
 "updateTime": 1660801717945,  
 "isWorking": true,  
 "workingTime": 1660801715639,  
 "origQuoteOrderQty": "0.00000000",  
 "selfTradePreventionMode": "NONE"  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 6  
 }  
 ]  
}

**Note:** The payload above does not show all fields that can appear. Please refer to [Conditional fields in Order Responses](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/trading-requests#conditional-fields-in-order-responses).

**Account order history (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-order-history-user_data)

{  
 "id": "734235c2-13d2-4574-be68-723e818c08f3",  
 "method": "allOrders",  
 "params": {  
 "symbol": "BTCUSDT",  
 "startTime": 1660780800000,  
 "endTime": 1660867200000,  
 "limit": 5,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "f50a972ba7fad92842187643f6b930802d4e20bce1ba1e788e856e811577bd42",  
 "timestamp": 1661955123341  
 }  
}

Query information about all your orders – active, canceled, filled – filtered by time range.

**Weight:** 20

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderId | INT | NO | Order ID to begin at |
| startTime | LONG | NO |  |
| endTime | LONG | NO |  |
| limit | INT | NO | Default: 500; Maximum: 1000 |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* If startTime and/or endTime are specified, orderId is ignored.

Orders are filtered by time of the last execution status update.

* If orderId is specified, return orders with order ID >= orderId.
* If no condition is specified, the most recent orders are returned.
* For some historical orders the cummulativeQuoteQty response field may be negative, meaning the data is not available at this time.
* The time between startTime and endTime can't be longer than 24 hours.

**Data Source:** Database

**Response:**

Status reports for orders are identical to [order.status](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-user_data).

Note that some fields are optional and included only for orders that set them.

{  
 "id": "734235c2-13d2-4574-be68-723e818c08f3",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "clientOrderId": "4d96324ff9d44481926157",  
 "price": "23416.10000000",  
 "origQty": "0.00847000",  
 "executedQty": "0.00847000",  
 "cummulativeQuoteQty": "198.33521500",  
 "status": "FILLED",  
 "timeInForce": "GTC",  
 "type": "LIMIT",  
 "side": "SELL",  
 "stopPrice": "0.00000000",  
 "icebergQty": "0.00000000",  
 "time": 1660801715639,  
 "updateTime": 1660801717945,  
 "isWorking": true,  
 "workingTime": 1660801715639,  
 "origQuoteOrderQty": "0.00000000",  
 "selfTradePreventionMode": "NONE",  
 "preventedMatchId": 0, *// This field only appears if the order expired due to STP.*  
 "preventedQuantity": "1.200000" *// This field only appears if the order expired due to STP.*  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Query Order list (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-list-user_data)

{  
 "id": "b53fd5ff-82c7-4a04-bd64-5f9dc42c2100",  
 "method": "orderList.status",  
 "params": {  
 "origClientOrderId": "08985fedd9ea2cf6b28996"  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "d12f4e8892d46c0ddfbd43d556ff6d818581b3be22a02810c2c20cb719aed6a4",  
 "timestamp": 1660801713965  
 }  
}

Check execution status of an Order list.

For execution status of individual orders, use [order.status](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-user_data).

**Weight:** 4

**Parameters**:

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| origClientOrderId | STRING | NO\* | Query order list by listClientOrderId. orderListId or origClientOrderId must be provided. |
| orderListId | INT | Query order list by orderListId. orderListId or origClientOrderId must be provided. |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* origClientOrderId refers to listClientOrderId of the order list itself.
* If both origClientOrderId and orderListId parameters are specified, only origClientOrderId is used and orderListId is ignored.

**Data Source:** Database

**Response:**

{  
 "id": "b53fd5ff-82c7-4a04-bd64-5f9dc42c2100",  
 "status": 200,  
 "result": {  
 "orderListId": 1274512,  
 "contingencyType": "OCO",  
 "listStatusType": "EXEC\_STARTED",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "08985fedd9ea2cf6b28996",  
 "transactionTime": 1660801713793,  
 "symbol": "BTCUSDT",  
 "orders": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138901,  
 "clientOrderId": "BqtFCj5odMoWtSqGk2X9tU"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138902,  
 "clientOrderId": "jLnZpj5enfMXTuhKB1d0us"  
 }  
 ]  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

**Current open order lists (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#current-open-order-lists-user_data)

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "method": "openOrderLists.status",  
 "params": {  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "1bea8b157dd78c3da30359bddcd999e4049749fe50b828e620e12f64e8b433c9",  
 "timestamp": 1660801713831  
 }  
}

Query execution status of all open order lists.

If you need to continuously monitor order status updates, please consider using WebSocket Streams:

* [userDataStream.start](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#user-data-stream-requests) request
* [executionReport](https://developers.binance.com/docs/binance-spot-api-docs/testnet/user-data-stream#order-update) user data stream event

**Weight**: 6

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

**Data Source:** Database

**Response:**

{  
 "id": "3a4437e2-41a3-4c19-897c-9cadc5dce8b6",  
 "status": 200,  
 "result": [  
 {  
 "orderListId": 0,  
 "contingencyType": "OCO",  
 "listStatusType": "EXEC\_STARTED",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "08985fedd9ea2cf6b28996",  
 "transactionTime": 1660801713793,  
 "symbol": "BTCUSDT",  
 "orders": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 4,  
 "clientOrderId": "CUhLgTXnX5n2c0gWiLpV4d"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 5,  
 "clientOrderId": "1ZqG7bBuYwaF4SU8CwnwHm"  
 }  
 ]  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 6  
 }  
 ]  
}

**Account order list history (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-order-list-history-user_data)

{  
 "id": "8617b7b3-1b3d-4dec-94cd-eefd929b8ceb",  
 "method": "allOrderLists",  
 "params": {  
 "startTime": 1660780800000,  
 "endTime": 1660867200000,  
 "limit": 5,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "c8e1484db4a4a02d0e84dfa627eb9b8298f07ebf12fcc4eaf86e4a565b2712c2",  
 "timestamp": 1661955123341  
 }  
}

Query information about all your order lists, filtered by time range.

**Weight:** 20

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| fromId | INT | NO | Order list ID to begin at |
| startTime | LONG | NO |  |
| endTime | LONG | NO |  |
| limit | INT | NO | Default: 500; Maximum: 1000 |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* If startTime and/or endTime are specified, fromId is ignored.

Order lists are filtered by transactionTime of the last order list execution status update.

* If fromId is specified, return order lists with order list ID >= fromId.
* If no condition is specified, the most recent order lists are returned.
* The time between startTime and endTime can't be longer than 24 hours.

**Data Source:** Database

**Response:**

Status reports for order lists are identical to [orderList.status](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-list-user_data).

{  
 "id": "8617b7b3-1b3d-4dec-94cd-eefd929b8ceb",  
 "status": 200,  
 "result": [  
 {  
 "orderListId": 1274512,  
 "contingencyType": "OCO",  
 "listStatusType": "EXEC\_STARTED",  
 "listOrderStatus": "EXECUTING",  
 "listClientOrderId": "08985fedd9ea2cf6b28996",  
 "transactionTime": 1660801713793,  
 "symbol": "BTCUSDT",  
 "orders": [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138901,  
 "clientOrderId": "BqtFCj5odMoWtSqGk2X9tU"  
 },  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 12569138902,  
 "clientOrderId": "jLnZpj5enfMXTuhKB1d0us"  
 }  
 ]  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Account trade history (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-trade-history-user_data)

{  
 "id": "f4ce6a53-a29d-4f70-823b-4ab59391d6e8",  
 "method": "myTrades",  
 "params": {  
 "symbol": "BTCUSDT",  
 "startTime": 1660780800000,  
 "endTime": 1660867200000,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "c5a5ffb79fd4f2e10a92f895d488943a57954edf5933bde3338dfb6ea6d6eefc",  
 "timestamp": 1661955125250  
 }  
}

Query information about all your trades, filtered by time range.

**Weight:**

| Condition | Weight |
| --- | --- |
| Without orderId | 20 |
| With orderId | 5 |

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderId | INT | NO |  |
| startTime | LONG | NO |  |
| endTime | LONG | NO |  |
| fromId | INT | NO | First trade ID to query |
| limit | INT | NO | Default: 500; Maximum: 1000 |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

Notes:

* If fromId is specified, return trades with trade ID >= fromId.
* If startTime and/or endTime are specified, trades are filtered by execution time (time).

fromId cannot be used together with startTime and endTime.

* If orderId is specified, only trades related to that order are returned.

startTime and endTime cannot be used together with orderId.

* If no condition is specified, the most recent trades are returned.
* The time between startTime and endTime can't be longer than 24 hours.

**Data Source:** Memory => Database

**Response:**

{  
 "id": "f4ce6a53-a29d-4f70-823b-4ab59391d6e8",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "id": 1650422481,  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "price": "23416.10000000",  
 "qty": "0.00635000",  
 "quoteQty": "148.69223500",  
 "commission": "0.00000000",  
 "commissionAsset": "BNB",  
 "time": 1660801715793,  
 "isBuyer": false,  
 "isMaker": true,  
 "isBestMatch": true  
 },  
 {  
 "symbol": "BTCUSDT",  
 "id": 1650422482,  
 "orderId": 12569099453,  
 "orderListId": -1,  
 "price": "23416.50000000",  
 "qty": "0.00212000",  
 "quoteQty": "49.64298000",  
 "commission": "0.00000000",  
 "commissionAsset": "BNB",  
 "time": 1660801715793,  
 "isBuyer": false,  
 "isMaker": true,  
 "isBestMatch": true  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Account unfilled order count (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-unfilled-order-count-user_data)

{  
 "id": "d3783d8d-f8d1-4d2c-b8a0-b7596af5a664",  
 "method": "account.rateLimits.orders",  
 "params": {  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "76289424d6e288f4dc47d167ac824e859dabf78736f4348abbbac848d719eb94",  
 "timestamp": 1660801839500  
 }  
}

Query your current unfilled order count for all intervals.

**Weight:** 40

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| apiKey | STRING | YES |  |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| signature | STRING | YES |  |
| timestamp | LONG | YES |  |

**Data Source:** Memory

**Response:**

{  
 "id": "d3783d8d-f8d1-4d2c-b8a0-b7596af5a664",  
 "status": 200,  
 "result": [  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "SECOND",  
 "intervalNum": 10,  
 "limit": 50,  
 "count": 0  
 },  
 {  
 "rateLimitType": "ORDERS",  
 "interval": "DAY",  
 "intervalNum": 1,  
 "limit": 160000,  
 "count": 0  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 40  
 }  
 ]  
}

**Account prevented matches (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-prevented-matches-user_data)

{  
 "id": "g4ce6a53-a39d-4f71-823b-4ab5r391d6y8",  
 "method": "myPreventedMatches",  
 "params": {  
 "symbol": "BTCUSDT",  
 "orderId": 35,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "c5a5ffb79fd4f2e10a92f895d488943a57954edf5933bde3338dfb6ea6d6eefc",  
 "timestamp": 1673923281052  
 }  
}

Displays the list of orders that were expired due to STP.

These are the combinations supported:

* symbol + preventedMatchId
* symbol + orderId
* symbol + orderId + fromPreventedMatchId (limit will default to 500)
* symbol + orderId + fromPreventedMatchId + limit

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| preventedMatchId | LONG | NO |  |
| orderId | LONG | NO |  |
| fromPreventedMatchId | LONG | NO |  |
| limit | INT | NO | Default: 500; Maximum: 1000 |
| recvWindow | LONG | NO | The value cannot be greater than 60000 |
| timestamp | LONG | YES |  |

**Weight**

| Case | Weight |
| --- | --- |
| If symbol is invalid | 2 |
| Querying by preventedMatchId | 2 |
| Querying by orderId | 20 |

**Data Source:**

Database

**Response:**

{  
 "id": "g4ce6a53-a39d-4f71-823b-4ab5r391d6y8",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "preventedMatchId": 1,  
 "takerOrderId": 5,  
 "makerSymbol": "BTCUSDT",  
 "makerOrderId": 3,  
 "tradeGroupId": 1,  
 "selfTradePreventionMode": "EXPIRE\_MAKER",  
 "price": "1.100000",  
 "makerPreventedQuantity": "1.300000",  
 "transactTime": 1669101687094  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Account allocations (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-allocations-user_data)

{  
 "id": "g4ce6a53-a39d-4f71-823b-4ab5r391d6y8",  
 "method": "myAllocations",  
 "params": {  
 "symbol": "BTCUSDT",  
 "orderId": 500,  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "c5a5ffb79fd4f2e10a92f895d488943a57954edf5933bde3338dfb6ea6d6eefc",  
 "timestamp": 1673923281052  
 }  
}

Retrieves allocations resulting from SOR order placement.

**Weight:** 20

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | Yes |  |
| startTime | LONG | No |  |
| endTime | LONG | No |  |
| fromAllocationId | INT | No |  |
| limit | INT | No | Default: 500; Maximum: 1000 |
| orderId | LONG | No |  |
| recvWindow | LONG | No | The value cannot be greater than 60000 |
| timestamp | LONG | No |  |

Supported parameter combinations:

| Parameters | Response |
| --- | --- |
| symbol | allocations from oldest to newest |
| symbol + startTime | oldest allocations since startTime |
| symbol + endTime | newest allocations until endTime |
| symbol + startTime + endTime | allocations within the time range |
| symbol + fromAllocationId | allocations by allocation ID |
| symbol + orderId | allocations related to an order starting with oldest |
| symbol + orderId + fromAllocationId | allocations related to an order by allocation ID |

**Note:** The time between startTime and endTime can't be longer than 24 hours.

**Data Source:** Database

**Response:**

{  
 "id": "g4ce6a53-a39d-4f71-823b-4ab5r391d6y8",  
 "status": 200,  
 "result": [  
 {  
 "symbol": "BTCUSDT",  
 "allocationId": 0,  
 "allocationType": "SOR",  
 "orderId": 500,  
 "orderListId": -1,  
 "price": "1.00000000",  
 "qty": "0.10000000",  
 "quoteQty": "0.10000000",  
 "commission": "0.00000000",  
 "commissionAsset": "BTC",  
 "time": 1687319487614,  
 "isBuyer": false,  
 "isMaker": false,  
 "isAllocator": false  
 }  
 ],  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Account Commission Rates (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#account-commission-rates-user_data)

{  
 "id": "d3df8a61-98ea-4fe0-8f4e-0fcea5d418b0",  
 "method": "account.commission",  
 "params": {  
 "symbol": "BTCUSDT",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A",  
 "signature": "c5a5ffb79fd4f2e10a92f895d488943a57954edf5933bde3338dfb6ea6d6eefc",  
 "timestamp": 1673923281052  
 }  
}

Get current account commission rates.

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |

**Weight:** 20

**Data Source:** Database

**Response:**

{  
 "id": "d3df8a61-98ea-4fe0-8f4e-0fcea5d418b0",  
 "status": 200,  
 "result": {  
 "symbol": "BTCUSDT",  
 "standardCommission": { *//Standard commission rates on trades from the order.*  
 "maker": "0.00000010",  
 "taker": "0.00000020",  
 "buyer": "0.00000030",  
 "seller": "0.00000040"  
 },  
 "taxCommission": { *//Tax commission rates on trades from the order.*  
 "maker": "0.00000112",  
 "taker": "0.00000114",  
 "buyer": "0.00000118",  
 "seller": "0.00000116"  
 },  
 "discount": { *//Discount on standard commissions when paying in BNB.*  
 "enabledForAccount": true,  
 "enabledForSymbol": true,  
 "discountAsset": "BNB",  
 "discount": "0.75000000" *//Standard commission is reduced by this rate when paying commission in BNB.*  
 }  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 20  
 }  
 ]  
}

**Query Order Amendments (USER\_DATA)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/account-requests#query-order-amendments-user_data)

{  
 "id": "6f5ebe91-01d9-43ac-be99-57cf062e0e30",  
 "method": "order.amendments",  
 "params": {  
 "orderId": "23",  
 "recvWindow": 5000,  
 "symbol": "BTCUSDT",  
 "timestamp": 1741925524887,  
 "apiKey": "N3Swv7WaBF7S2rzA12UkPunM3udJiDddbgv1W7CzFGnsQXH9H62zzSCST0CndjeE",  
 "signature": "0eed2e9d95b6868ea5ec21da0d14538192ef344c30ecf9fe83d58631699334dc"  
 }  
}

Queries all amendments of a single order.

**Weight**: 4

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| symbol | STRING | YES |  |
| orderId | LONG | YES |  |
| fromExecutionId | LONG | NO |  |
| limit | INT | NO | Default:500; Maximum: 1000 |
| recvWindow | LONG | NO | The value cannot be greater than 60000. |
| timestamp | LONG | YES |  |

**Data Source:** Database

**Response:**

{  
 "id": "6f5ebe91-01d9-43ac-be99-57cf062e0e30",  
 "status": 200,  
 "result":  
 [  
 {  
 "symbol": "BTCUSDT",  
 "orderId": 23,  
 "executionId": 60,  
 "origClientOrderId": "my\_pending\_order",  
 "newClientOrderId": "xbxXh5SSwaHS7oUEOCI88B",  
 "origQty": "7.00000000",  
 "newQty": "5.00000000",  
 "time": 1741924229819  
 }  
 ],  
 "rateLimits":  
 [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 4  
 }  
 ]  
}

**User Data Stream requests**

**User Data Stream subscription**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#user-data-stream-subscription)

**Subscribe to User Data Stream (USER\_STREAM)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#subscribe-to-user-data-stream-user_stream)

{  
 "id": "d3df8a21-98ea-4fe0-8f4e-0fcea5d418b7",  
 "method": "userDataStream.subscribe"  
}

Subscribe to the User Data Stream in the current WebSocket connection.

**Notes:**

* This method requires an authenticated WebSocket connection using Ed25519 keys. Please refer to [session.logon](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/authentication-requests#session-logon).
* To check the subscription status, use [session.status](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/authentication-requests#session-status), see the userDataStream flag indicating you have have an active subscription.
* User Data Stream events are available in both JSON and [SBE](https://developers.binance.com/docs/binance-spot-api-docs/faqs/sbe_faq) sessions.
  + Please refer to [User Data Streams](https://developers.binance.com/docs/binance-spot-api-docs/testnet/user-data-stream) for the event format details.
  + For SBE, only SBE schema 2:1 or later is supported.

**Weight**: 2

**Parameters**: NONE

**Response**:

{  
 "id": "d3df8a21-98ea-4fe0-8f4e-0fcea5d418b7",  
 "status": 200,  
 "result": {}  
}

**Unsubscribe from User Data Stream (USER\_STREAM)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#unsubscribe-from-user-data-stream-user_stream)

{  
 "id": "d3df8a21-98ea-4fe0-8f4e-0fcea5d418b7",  
 "method": "userDataStream.unsubscribe"  
}

Stop listening to the User Data Stream in the current WebSocket connection.

**Weight**: 2

**Parameters**:

NONE

**Response**:

{  
 "id": "d3df8a21-98ea-4fe0-8f4e-0fcea5d418b7",  
 "status": 200,  
 "result": {}  
}

**Listen Key Management (Deprecated)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#listen-key-management-deprecated)

[!IMPORTANT] These requests have been deprecated, which means we will remove them in the future. Please subscribe to the User Data Stream through the [WebSocket API](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#user_data_stream_subscribe) instead.

The following requests manage [User Data Stream](https://developers.binance.com/docs/binance-spot-api-docs/testnet/user-data-stream) subscriptions.

**Start user data stream (USER\_STREAM) (Deprecated)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#start-user-data-stream-user_stream-deprecated)

{  
 "id": "d3df8a61-98ea-4fe0-8f4e-0fcea5d418b0",  
 "method": "userDataStream.start",  
 "params": {  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A"  
 }  
}

Start a new user data stream.

**Note:** the stream will close in 60 minutes unless [userDataStream.ping](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#ping-user-data-stream-user_stream) requests are sent regularly.

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| apiKey | STRING | YES |  |

**Data Source:** Memory

**Response:**

Subscribe to the received listen key on WebSocket Stream afterwards.

{  
 "id": "d3df8a61-98ea-4fe0-8f4e-0fcea5d418b0",  
 "status": 200,  
 "result": {  
 "listenKey": "xs0mRXdAKlIPDRFrlPcw0qI41Eh3ixNntmymGyhrhgqo7L6FuLaWArTD7RLP"  
 },  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Ping user data stream (USER\_STREAM) (Deprecated)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#ping-user-data-stream-user_stream-deprecated)

{  
 "id": "815d5fce-0880-4287-a567-80badf004c74",  
 "method": "userDataStream.ping",  
 "params": {  
 "listenKey": "xs0mRXdAKlIPDRFrlPcw0qI41Eh3ixNntmymGyhrhgqo7L6FuLaWArTD7RLP",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A"  
 }  
}

Ping a user data stream to keep it alive.

User data streams close automatically after 60 minutes, even if you're listening to them on WebSocket Streams. In order to keep the stream open, you have to regularly send pings using the userDataStream.ping request.

It is recommended to send a ping once every 30 minutes.

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| listenKey | STRING | YES |  |
| apiKey | STRING | YES |  |

**Data Source:** Memory

**Response:**

{  
 "id": "815d5fce-0880-4287-a567-80badf004c74",  
 "status": 200,  
 "response": {},  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**Stop user data stream (USER\_STREAM) (Deprecated)**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/websocket-api/user-data-stream-requests#stop-user-data-stream-user_stream-deprecated)

{  
 "id": "819e1b1b-8c06-485b-a13e-131326c69599",  
 "method": "userDataStream.stop",  
 "params": {  
 "listenKey": "xs0mRXdAKlIPDRFrlPcw0qI41Eh3ixNntmymGyhrhgqo7L6FuLaWArTD7RLP",  
 "apiKey": "vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A"  
 }  
}

Explicitly stop and close the user data stream.

**Weight:** 2

**Parameters:**

| Name | Type | Mandatory | Description |
| --- | --- | --- | --- |
| listenKey | STRING | YES |  |
| apiKey | STRING | YES |  |

**Data Source:** Memory

**Response:**

{  
 "id": "819e1b1b-8c06-485b-a13e-131326c69599",  
 "status": 200,  
 "response": {},  
 "rateLimits": [  
 {  
 "rateLimitType": "REQUEST\_WEIGHT",  
 "interval": "MINUTE",  
 "intervalNum": 1,  
 "limit": 6000,  
 "count": 2  
 }  
 ]  
}

**WebSocket Streams for Binance SPOT Testnet**

**Last Updated: 2025-04-01**

**General WSS information**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#general-wss-information)

* The base endpoint is: **wss://stream.testnet.binance.vision/ws**.
* Streams can be accessed either in a single raw stream or in a combined stream
* Raw streams are accessed at **/ws/<streamName>**
* Combined streams are accessed at **/stream?streams=<streamName1>/<streamName2>/<streamName3>**
* Combined stream events are wrapped as follows: **{"stream":"<streamName>","data":<rawPayload>}**
* All symbols for streams are **lowercase**
* All time and timestamp related fields are **milliseconds by default**. To receive the information in microseconds, please add the parameter timeUnit=MICROSECOND or timeUnit=microsecond in the URL.
  + For example: /stream?streams=btcusdt@trade&timeUnit=MICROSECOND
* A single connection to **stream.binance.com** is only valid for 24 hours; expect to be disconnected at the 24 hour mark
* The WebSocket server will send a ping frame every 20 seconds.
  + If the WebSocket server does not receive a pong frame back from the connection within a minute, the connection will be disconnected.
  + When you receive a ping, you must send a pong with a copy of ping's payload as soon as possible.
  + Unsolicited pong frames are allowed, but will not prevent disconnection. **It is recommended that the payload for these pong frames are empty.**
* The base endpoint **wss://data-stream.binance.vision** can be subscribed to receive **only** market data messages.  
  User data stream is **NOT** available from this URL.
* All time and timestamp related fields are **milliseconds by default**. To receive the information in microseconds, please add the parameter timeUnit=MICROSECOND or timeUnit=microsecond in the URL.
  + For example: /stream?streams=btcusdt@trade&timeUnit=MICROSECOND

**WebSocket Limits**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#websocket-limits)

* WebSocket connections have a limit of 5 incoming messages per second. A message is considered:
  + A PING frame
  + A PONG frame
  + A JSON controlled message (e.g. subscribe, unsubscribe)
* A connection that goes beyond the limit will be disconnected; IPs that are repeatedly disconnected may be banned.
* A single connection can listen to a maximum of 1024 streams.
* There is a limit of **300 connections per attempt every 5 minutes per IP**.

**Live Subscribing/Unsubscribing to streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#live-subscribingunsubscribing-to-streams)

* The following data can be sent through the WebSocket instance in order to subscribe/unsubscribe from streams. Examples can be seen below.
* The id is used as an identifier to uniquely identify the messages going back and forth. The following formats are accepted:
  + 64-bit signed integer
  + alphanumeric strings; max length 36
  + null
* In the response, if the result received is null this means the request sent was a success for non-query requests (e.g. Subscribing/Unsubscribing).

**Subscribe to a stream**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#subscribe-to-a-stream)

* Request

{  
 "method": "SUBSCRIBE",  
 "params": [  
 "btcusdt@aggTrade",  
 "btcusdt@depth"  
 ],  
 "id": 1  
}

* Response

{  
 "result": null,  
 "id": 1  
}

**Unsubscribe to a stream**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#unsubscribe-to-a-stream)

* Request

{  
 "method": "UNSUBSCRIBE",  
 "params": [  
 "btcusdt@depth"  
 ],  
 "id": 312  
}

* Response

{  
 "result": null,  
 "id": 312  
}

**Listing Subscriptions**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#listing-subscriptions)

* Request

{  
 "method": "LIST\_SUBSCRIPTIONS",  
 "id": 3  
}

* Response

{  
 "result": [  
 "btcusdt@aggTrade"  
 ],  
 "id": 3  
}

**Setting Properties**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#setting-properties)

Currently, the only property that can be set is whether combined stream payloads are enabled or not. The combined property is set to false when connecting using /ws/ ("raw streams") and true when connecting using /stream/.

* Request

{  
 "method": "SET\_PROPERTY",  
 "params": [  
 "combined",  
 true  
 ],  
 "id": 5  
}

* Response

{  
 "result": null,  
 "id": 5  
}

**Retrieving Properties**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#retrieving-properties)

* Request

{  
 "method": "GET\_PROPERTY",  
 "params": [  
 "combined"  
 ],  
 "id": 2  
}

* Response

{  
 "result": true, *// Indicates that combined is set to true.*  
 "id": 2  
}

**Error Messages**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#error-messages)

| Error Message | Description |
| --- | --- |
| {"code": 0, "msg": "Unknown property","id": %s} | Parameter used in the SET\_PROPERTY or GET\_PROPERTY was invalid |
| {"code": 1, "msg": "Invalid value type: expected Boolean"} | Value should only be true or false |
| {"code": 2, "msg": "Invalid request: property name must be a string"} | Property name provided was invalid |
| {"code": 2, "msg": "Invalid request: request ID must be an unsigned integer"} | Parameter id had to be provided or the value provided in the id parameter is an unsupported type |
| {"code": 2, "msg": "Invalid request: unknown variant %s, expected one of SUBSCRIBE, UNSUBSCRIBE, LIST\_SUBSCRIPTIONS, SET\_PROPERTY, GET\_PROPERTY at line 1 column 28"} | Possible typo in the provided method or provided method was neither of the expected values |
| {"code": 2, "msg": "Invalid request: too many parameters"} | Unnecessary parameters provided in the data |
| {"code": 2, "msg": "Invalid request: property name must be a string"} | Property name was not provided |
| {"code": 2, "msg": "Invalid request: missing field method at line 1 column 73"} | method was not provided in the data |
| {"code":3,"msg":"Invalid JSON: expected value at line %s column %s"} | JSON data sent has incorrect syntax. |

**Detailed Stream information**

**Aggregate Trade Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#aggregate-trade-streams)

The Aggregate Trade Streams push trade information that is aggregated for a single taker order.

**Stream Name:** <symbol>@aggTrade

**Update Speed:** Real-time

**Payload:**

{  
 "e": "aggTrade", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "a": 12345, *// Aggregate trade ID*  
 "p": "0.001", *// Price*  
 "q": "100", *// Quantity*  
 "f": 100, *// First trade ID*  
 "l": 105, *// Last trade ID*  
 "T": 1672515782136, *// Trade time*  
 "m": true, *// Is the buyer the market maker?*  
 "M": true *// Ignore*  
}

**Trade Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#trade-streams)

The Trade Streams push raw trade information; each trade has a unique buyer and seller.

**Stream Name:** <symbol>@trade

**Update Speed:** Real-time

**Payload:**

{  
 "e": "trade", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "t": 12345, *// Trade ID*  
 "p": "0.001", *// Price*  
 "q": "100", *// Quantity*  
 "T": 1672515782136, *// Trade time*  
 "m": true, *// Is the buyer the market maker?*  
 "M": true *// Ignore*  
}

**Kline/Candlestick Streams for UTC**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#klinecandlestick-streams-for-utc)

The Kline/Candlestick Stream push updates to the current klines/candlestick every second in UTC+0 timezone.

**Kline/Candlestick chart intervals:**

s-> seconds; m -> minutes; h -> hours; d -> days; w -> weeks; M -> months

* 1s
* 1m
* 3m
* 5m
* 15m
* 30m
* 1h
* 2h
* 4h
* 6h
* 8h
* 12h
* 1d
* 3d
* 1w
* 1M

**Stream Name:** <symbol>@kline\_<interval>

**Update Speed:** 1000ms for 1s, 2000ms for the other intervals

**Payload:**

{  
 "e": "kline", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "k": {  
 "t": 1672515780000, *// Kline start time*  
 "T": 1672515839999, *// Kline close time*  
 "s": "BNBBTC", *// Symbol*  
 "i": "1m", *// Interval*  
 "f": 100, *// First trade ID*  
 "L": 200, *// Last trade ID*  
 "o": "0.0010", *// Open price*  
 "c": "0.0020", *// Close price*  
 "h": "0.0025", *// High price*  
 "l": "0.0015", *// Low price*  
 "v": "1000", *// Base asset volume*  
 "n": 100, *// Number of trades*  
 "x": false, *// Is this kline closed?*  
 "q": "1.0000", *// Quote asset volume*  
 "V": "500", *// Taker buy base asset volume*  
 "Q": "0.500", *// Taker buy quote asset volume*  
 "B": "123456" *// Ignore*  
 }  
}

**Kline/Candlestick Streams with timezone offset**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#klinecandlestick-streams-with-timezone-offset)

The Kline/Candlestick Stream push updates to the current klines/candlestick every second in UTC+8 timezone.

**Kline/Candlestick chart intervals:** Supported intervals: See [Kline/Candlestick chart intervals](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#kline-intervals)

**UTC+8 timezone offset:**

* Kline intervals open and close in the UTC+8 timezone. For example the 1d klines will open at the beginning of the UTC+8 day, and close at the end of the UTC+8 day.
* Note that E (event time), t (start time), and T (close time) in the payload are Unix timestamps, which are always interpreted in UTC.

**Stream Name:** <symbol>@kline\_<interval>@+08:00

**Update Speed:** 1000ms for 1s, 2000ms for the other intervals

**Payload:**

{  
 "e": "kline", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "k": {  
 "t": 1672515780000, *// Kline start time*  
 "T": 1672515839999, *// Kline close time*  
 "s": "BNBBTC", *// Symbol*  
 "i": "1m", *// Interval*  
 "f": 100, *// First trade ID*  
 "L": 200, *// Last trade ID*  
 "o": "0.0010", *// Open price*  
 "c": "0.0020", *// Close price*  
 "h": "0.0025", *// High price*  
 "l": "0.0015", *// Low price*  
 "v": "1000", *// Base asset volume*  
 "n": 100, *// Number of trades*  
 "x": false, *// Is this kline closed?*  
 "q": "1.0000", *// Quote asset volume*  
 "V": "500", *// Taker buy base asset volume*  
 "Q": "0.500", *// Taker buy quote asset volume*  
 "B": "123456" *// Ignore*  
 }  
}

**Individual Symbol Mini Ticker Stream**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-mini-ticker-stream)

24hr rolling window mini-ticker statistics. These are NOT the statistics of the UTC day, but a 24hr rolling window for the previous 24hrs.

**Stream Name:** <symbol>@miniTicker

**Update Speed:** 1000ms

**Payload:**

{  
 "e": "24hrMiniTicker", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "c": "0.0025", *// Close price*  
 "o": "0.0010", *// Open price*  
 "h": "0.0025", *// High price*  
 "l": "0.0010", *// Low price*  
 "v": "10000", *// Total traded base asset volume*  
 "q": "18" *// Total traded quote asset volume*  
 }

**All Market Mini Tickers Stream**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#all-market-mini-tickers-stream)

24hr rolling window mini-ticker statistics for all symbols that changed in an array. These are NOT the statistics of the UTC day, but a 24hr rolling window for the previous 24hrs. Note that only tickers that have changed will be present in the array.

**Stream Name:** !miniTicker@arr

**Update Speed:** 1000ms

**Payload:**

[  
 {  
 *// Same as <symbol>@miniTicker payload*  
 }  
]

**Individual Symbol Ticker Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-ticker-streams)

24hr rolling window ticker statistics for a single symbol. These are NOT the statistics of the UTC day, but a 24hr rolling window for the previous 24hrs.

**Stream Name:** <symbol>@ticker

**Update Speed:** 1000ms

**Payload:**

{  
 "e": "24hrTicker", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "p": "0.0015", *// Price change*  
 "P": "250.00", *// Price change percent*  
 "w": "0.0018", *// Weighted average price*  
 "x": "0.0009", *// First trade(F)-1 price (first trade before the 24hr rolling window)*  
 "c": "0.0025", *// Last price*  
 "Q": "10", *// Last quantity*  
 "b": "0.0024", *// Best bid price*  
 "B": "10", *// Best bid quantity*  
 "a": "0.0026", *// Best ask price*  
 "A": "100", *// Best ask quantity*  
 "o": "0.0010", *// Open price*  
 "h": "0.0025", *// High price*  
 "l": "0.0010", *// Low price*  
 "v": "10000", *// Total traded base asset volume*  
 "q": "18", *// Total traded quote asset volume*  
 "O": 0, *// Statistics open time*  
 "C": 86400000, *// Statistics close time*  
 "F": 0, *// First trade ID*  
 "L": 18150, *// Last trade Id*  
 "n": 18151 *// Total number of trades*  
}

**All Market Tickers Stream**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#all-market-tickers-stream)

24hr rolling window ticker statistics for all symbols that changed in an array. These are NOT the statistics of the UTC day, but a 24hr rolling window for the previous 24hrs. Note that only tickers that have changed will be present in the array.

**Stream Name:** !ticker@arr

**Update Speed:** 1000ms

**Payload:**

[  
 {  
 *// Same as <symbol>@ticker payload*  
 }  
]

**Individual Symbol Rolling Window Statistics Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-rolling-window-statistics-streams)

Rolling window ticker statistics for a single symbol, computed over multiple windows.

**Stream Name:** <symbol>@ticker\_<window\_size>

**Window Sizes:** 1h,4h,1d

**Update Speed:** 1000ms

**Note**: This stream is different from the <symbol>@ticker stream. The open time O always starts on a minute, while the closing time C is the current time of the update. As such, the effective window might be up to 59999ms wider that <window\_size>.

**Payload:**

{  
 "e": "1hTicker", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "p": "0.0015", *// Price change*  
 "P": "250.00", *// Price change percent*  
 "o": "0.0010", *// Open price*  
 "h": "0.0025", *// High price*  
 "l": "0.0010", *// Low price*  
 "c": "0.0025", *// Last price*  
 "w": "0.0018", *// Weighted average price*  
 "v": "10000", *// Total traded base asset volume*  
 "q": "18", *// Total traded quote asset volume*  
 "O": 0, *// Statistics open time*  
 "C": 1675216573749, *// Statistics close time*  
 "F": 0, *// First trade ID*  
 "L": 18150, *// Last trade Id*  
 "n": 18151 *// Total number of trades*  
}

**All Market Rolling Window Statistics Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#all-market-rolling-window-statistics-streams)

Rolling window ticker statistics for all market symbols, computed over multiple windows. Note that only tickers that have changed will be present in the array.

**Stream Name:** !ticker\_<window-size>@arr

**Window Size:** 1h,4h,1d

**Update Speed:** 1000ms

**Payload:**

[  
 {  
 *// Same as <symbol>@ticker\_<window-size> payload,*  
 *// one for each symbol updated within the interval.*  
 }  
]

**Individual Symbol Book Ticker Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#individual-symbol-book-ticker-streams)

Pushes any update to the best bid or ask's price or quantity in real-time for a specified symbol. Multiple <symbol>@bookTicker streams can be subscribed to over one connection.

**Stream Name:** <symbol>@bookTicker

**Update Speed:** Real-time

**Payload:**

{  
 "u":400900217, *// order book updateId*  
 "s":"BNBUSDT", *// symbol*  
 "b":"25.35190000", *// best bid price*  
 "B":"31.21000000", *// best bid qty*  
 "a":"25.36520000", *// best ask price*  
 "A":"40.66000000" *// best ask qty*  
}

**Average Price**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#average-price)

Average price streams push changes in the average price over a fixed time interval.

**Stream Name:** <symbol>@avgPrice

**Update Speed:** 1000ms

**Payload:**

{  
 "e": "avgPrice", *// Event type*  
 "E": 1693907033000, *// Event time*  
 "s": "BTCUSDT", *// Symbol*  
 "i": "5m", *// Average price interval*  
 "w": "25776.86000000", *// Average price*  
 "T": 1693907032213 *// Last trade time*  
}

**Partial Book Depth Streams**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#partial-book-depth-streams)

Top **<levels>** bids and asks, pushed every second. Valid **<levels>** are 5, 10, or 20.

**Stream Names:** <symbol>@depth<levels> OR <symbol>@depth<levels>@100ms

**Update Speed:** 1000ms or 100ms

**Payload:**

{  
 "lastUpdateId": 160, *// Last update ID*  
 "bids": [ *// Bids to be updated*  
 [  
 "0.0024", *// Price level to be updated*  
 "10" *// Quantity*  
 ]  
 ],  
 "asks": [ *// Asks to be updated*  
 [  
 "0.0026", *// Price level to be updated*  
 "100" *// Quantity*  
 ]  
 ]  
}

**Diff. Depth Stream**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#diff-depth-stream)

Order book price and quantity depth updates used to locally manage an order book.

**Stream Name:** <symbol>@depth OR <symbol>@depth@100ms

**Update Speed:** 1000ms or 100ms

**Payload:**

{  
 "e": "depthUpdate", *// Event type*  
 "E": 1672515782136, *// Event time*  
 "s": "BNBBTC", *// Symbol*  
 "U": 157, *// First update ID in event*  
 "u": 160, *// Final update ID in event*  
 "b": [ *// Bids to be updated*  
 [  
 "0.0024", *// Price level to be updated*  
 "10" *// Quantity*  
 ]  
 ],  
 "a": [ *// Asks to be updated*  
 [  
 "0.0026", *// Price level to be updated*  
 "100" *// Quantity*  
 ]  
 ]  
}

**How to manage a local order book correctly**[**​**](https://developers.binance.com/docs/binance-spot-api-docs/testnet/web-socket-streams#how-to-manage-a-local-order-book-correctly)

1. Open a WebSocket connection to wss://stream.testnet.binance.vision:9443/ws/bnbbtc@depth.
2. Buffer the events received from the stream. Note the U of the first event you received.
3. Get a depth snapshot from https://testnet.binance.vision/api/v3/depth?symbol=BNBBTC&limit=5000.
4. If the lastUpdateId from the snapshot is strictly less than the U from step 2, go back to step 3.
5. In the buffered events, discard any event where u is <= lastUpdateId of the snapshot. The first buffered event should now have lastUpdateId within its [U;u] range.
6. Set your local order book to the snapshot. Its update ID is lastUpdateId.
7. Apply the update procedure below to all buffered events, and then to all subsequent events received.

To apply an event to your local order book, follow this update procedure:

1. If the event u (last update ID) is < the update ID of your local order book, ignore the event.
2. If the event U (first update ID) is > the update ID of your local order book, something went wrong. Discard your local order book and restart the process from the beginning.
3. For each price level in bids (b) and asks (a), set the new quantity in the order book:
   * If the price level does not exist in the order book, insert it with new quantity.
   * If the quantity is zero, remove the price level from the order book.
4. Set the order book update ID to the last update ID (u) in the processed event.

[!NOTE] Since depth snapshots retrieved from the API have a limit on the number of price levels (5000 on each side maximum), you won't learn the quantities for the levels outside of the initial snapshot unless they change.  
So be careful when using the information for those levels, since they might not reflect the full view of the order book.  
However, for most use cases, seeing 5000 levels on each side is enough to understand the market and trade effectively.