



现货/杠杆/币安宝/矿池 U本位合约 币本位合约 欧式期权

[English](#)

- [更新日志](#)
- [基本信息](#)
 - [SDK和代码示例](#)
 - [Rest 基本信息](#)
 - [testnet](#)
 - [访问限制](#)
 - [接口鉴权类型](#)
 - [需要签名的接口 \(TRADE 与 USER DATA\)](#)
 - [公开API参数](#)
 - [过滤器](#)
 - [Postman Collections](#)
- [行情接口](#)
 - [测试服务器连通性 PING](#)
 - [获取服务器时间](#)
 - [获取交易规则和交易对](#)
 - [深度信息](#)
 - [近期成交](#)
 - [查询历史成交\(MARKET DATA\)](#)
 - [近期成交\(归集\)](#)
 - [K线数据](#)
 - [连续合约K线数据](#)
 - [价格指数K线数据](#)
 - [标记价格K线数据](#)
 - [最新标记价格和资金费率](#)
 - [查询资金费率历史](#)
 - [24hr价格变动情况](#)
 - [最新价格](#)
 - [当前最优挂单](#)
 - [获取未平仓合约数](#)
 - [合约持仓量](#)
 - [大户账户数多空比](#)
 - [大户持仓量多空比](#)
 - [多空持仓人数比](#)
 - [合约主动买卖量](#)
 - [杠杆代币历史净值K线](#)
 - [综合指数交易对信息](#)
 - [多资产模式资产汇率指数](#)
- [Websocket 行情推送](#)
 - [实时订阅/取消数据流](#)
 - [最新合约价格](#)
 - [归集交易](#)
 - [最新标记价格](#)
 - [全市场最新标记价格](#)
 - [K线](#)
 - [连续合约K线](#)
 - [按Symbol的精简Ticker](#)
 - [全市场的精简Ticker](#)
 - [按Symbol的完整Ticker](#)

- 全市场的完整Ticker
- 按Symbol的最优挂单信息
- 全市场最优挂单信息
- 强平订单
- 全市场强平订单
- 有限档深度信息
- 增量深度信息
- 如何正确在本地维护一个orderbook副本
- 综合指数交易对信息流
- 账户和交易接口
 - 划转
 - 获取划转历史
 - 更改持仓模式(TRADE)
 - 查询持仓模式(USER DATA)
 - 更改联合保证金模式(TRADE)
 - 查询联合保证金模式(USER DATA)
 - 下单(TRADE)
 - 测试下单接口(TRADE)
 - 批量下单(TRADE)
 - 查询订单(USER DATA)
 - 撤销订单(TRADE)
 - 撤销全部订单(TRADE)
 - 批量撤销订单(TRADE)
 - 倒计时撤销所有订单(TRADE)
 - 查询当前挂单(USER DATA)
 - 查看当前全部挂单(USER DATA)
 - 查询所有订单(包括历史订单)(USER DATA)
 - 账户余额V2(USER DATA)
 - 账户信息V2(USER DATA)
 - 调整开仓杠杆(TRADE)
 - 变换逐全仓模式(TRADE)
 - 调整逐仓保证金(TRADE)
 - 逐仓保证金变动历史(TRADE)
 - 用户持仓风险V2(USER DATA)
 - 账户成交历史(USER DATA)
 - 获取账户损益资金流水(USER DATA)
 - 杠杆分层标准(USER DATA)
 - 持仓ADL队列估算(USER DATA)
 - 用户强平单历史(USER DATA)
 - 合约交易量化规则指标(USER DATA)
 - 用户手续费率(USER DATA)
 - 获取合约资金流水下载Id(USER DATA)
 - 通过下载Id获取合约资金流水下载链接(USER DATA)
- Websocket 账户信息推送
 - 生成listenKey(USER STREAM)
 - 延长listenKey有效期(USER STREAM)
 - 关闭listenKey(USER STREAM)
 - listenKey过期推送
 - 追加保证金通知
 - Balance和Position更新推送
 - 订单/交易更新推送
 - 杠杆倍数等账户配置更新推送
- 统一账户接口
 - 获取统一账户交易规则
 - 查询统一账户账户信息(USER DATA)

- **错误代码**

- [10xx - 常规服务器或网络问题](#)
- [11xx - Request issues](#)
- [20xx - Processing Issues](#)
- [40xx - Filters and other Issues](#)

更新日志

2022-10-13

注意: 此变动会在 2022-10-17 生效

REST RATE LIMIT WEIGHT

接口 `GET /fapi/v1/ticker/bookTicker`

权重更新:

- 单交易对 2
- 无交易对 5

2022-09-22

- 更新账户和交易接口：
 - `GET /fapi/v1/income` : 支持更多收益类型
- 新增统一账户接口：
 - `GET /fapi/v1/pmAccountInfo` : 查询统一账户当前账户信息。

2022-07-27

REST RATE LIMIT WEIGHT

- 接口 `GET /fapi/v1/trades` 的请求权重更新为 5

2022-06-28

REST

- 新增接口 `GET /fapi/v1/pmExchangeInfo` 获取统一账户交易规则

2022-04-08

WEBSOCKET

- 更换 base url `wss://nbstream.binance.com/1vt-p` 对于杠杆代币数据流 `<tokenName>@tokenNav` 和 `<tokenName>@nav_kline_<interval>`, 详情见: [Websocket 杠杆代币信息更新](#) 和 [Websocket 杠杆代币净值K线更新](#)

2022-03-01

REST

- 新增接口 `GET /fapi/v1/income/async` 获取合约资金流水下载id
- 新增接口 `GET /fapi/v1/income/async/id` 通过下载id获取合约资金流水下载链接

2022-02-10

REST

- 更新 `GET /fapi/v2/account` 接口：
 - 若用户开启多资产模式，`totalInitialMargin`, `totalMaintMargin`, `totalWalletBalance`, `totalUnrealizedProfit`, `totalMarginBalance`, `totalPositionInitialMargin`, `totalOpenOrderInitialMargin`, `totalCrossWalletBalance`, `totalCrossUnPnl`, `availableBalance`, `maxWithdrawAmount` 计入各种资产并转化为其USD价值显示
 - 若用户使用单资产模式，仅USDT资产会被计入计算（和改动前一致）

2021-12-30

WEBSOCKET

- 新增 WEBSOCKET 连接方式：
 - Base Url : `wss://fstream-auth.binance.com`
 - 订阅单一stream格式为 `/ws/<streamName>?listenKey=<validateListenKey>`
 - 组合streams的URL格式为 `/stream?streams=<streamName1>/<streamName2>/<streamName3>&listenKey=<validateListenKey>`
 - `<validateListenKey>` 在建立连接时，必须为一个有效的listenKey
- 详细说明见 Websocket 行情推送和Websocket 账户信息推送

2021-11-02

REST

- 新增接口 `GET /fapi/v1/assetIndex` 以获取多资产模式保证金资产汇率指数

2021-07-06

REST

- `GET /fapi/v2/account` 和 `GET /fapi/v2/positionRisk` 响应内容加入 `updateTime` 以表示资产，仓位的最新更新时间
- `GET /fapi/v1/exchangeInfo` 响应内容增加以下字段：
 - "liquidationFee" 表示强平费率
 - "marketTakeBound" 表示市价吃单(相对于标记价格)允许可造成的价格偏离比例

2021-06-15

WEBSOCKET

- 综合指数交易对信息流 `<symbol>@compositeIndex` 新增返回字段 "q" 表示报价资产， "i" 表示指数价格

REST

- 更新以下接口：
 - `GET /fapi/v1/indexInfo` 响应加入 `component` 成分资产，`quoteAsset` 报价资产字段

2021-05-06

WEBSOCKET

- 更新以下接口：
 - 原有杠杆倍数更新推送事件 `ACCOUNT_CONFIG_UPDATE` 扩展为账户配置更新推送事件，包含杠杆倍数与联合保证金状态更新推送
 - Balance和Position更新推送 `ACCOUNT_UPDATE` 的事件枚举类型新增 `AUTO_EXCHANGE` 代表联合保证金自动兑换事件

REST

- 新增以下接口:

- POST /fapi/v1/multiAssetsMargin 以更改联合保证金模式
- GET /fapi/v1/multiAssetsMargin 以查询联合保证金模式

- 更新以下接口:

- GET /fapi/v1/exchangeInfo 响应加入 assets 资产信息
- GET /fapi/v2/balance 与 GET /fapi/v2/account 响应加入 marginAvailable 字段代表是否可用作联合保证金

2021-04-27

WEBSOCKET

- 以下市场强平订单推送事件由实时推送调整为快照推送，即每秒最多推送一条强平订单数据:
 - <symbol>@forceOrder
 - !forceOrder@arr

REST

- 获取市场强平订单接口 GET /fapi/v1/allForceOrders 停止维护，不再接受请求

2021-04-22

WEBSOCKET

- "ACCOUNT_UPDATE" 事件新增返回字段 "bc" 表示账户余额改变量。

2021-03-02

- 新增接口 GET /fapi/v1/indexPriceKlines 以获取价格指数K线数据。
- 新增接口 GET /fapi/v1/markPriceKlines 以获取标记价格K线数据。

2021-02-24

REST RATE LIMIT WEIGHT

- 接口 GET /fapi/v2/balance 的请求权重更新为 5
- 接口 GET /fapi/v2/positionRisk 的请求权重更新为 5

2021-02-22

REST RATE LIMIT WEIGHT

- 接口 GET /fapi/v1/income 的请求权重更新为 30

REST

- 接口 GET /fapi/v1/allOrders 的查询时间范围最大为7天。
- 接口 GET /fapi/v1/allForceOrders 的查询范围仅限于最近7天内的数据。

2021-01-26

WEB SOCKET USER DATA STREAM

- USER-DATA-STREAM 中新增事件 ACCOUNT_CONFIG_UPDATE 以获取交易对杠杆倍数变动更新

REST RATE LIMIT WEIGHT

- 以下接口的权重调整为 带symbol 20, 不带symbol 50
 - GET /fapi/v1/allForceOrders
 - GET /fapi/v1/forceOrders

REST

- 新增交易对过滤器 "MIN_NOTIONAL"，定义了交易对订单所允许的最小名义价值，并在 fapi/v1/exchangeInfo 的响应中返回

2021-01-21

合约订单用户自定义id newClientOrderId 更新正则规则为: ^[\.A-Z\:/a-z0-9_-]{1,36} \$

2021-01-04

REST

- 以下接口的IP限制权重将采用基于参数LIMIT数值的新权重规则:
 - GET /fapi/v1/klines
 - GET /fapi/v1/continuousKlines
- 以下接口的IP限制权重调整到 20:
 - GET /fapi/v1/historicalTrades
 - GET /fapi/v1/allForceOrders
 - GET /fapi/v1/forceOrders
 - GET /fapi/v1/aggTrades

2020-12-08

WEBSOCKET

- 行情消息推送 <symbol>@bookTicker 和 !bookTicker 返回内容新增字段 e 表示事件类型
- 行情消息推送 <symbol>@markPrice, <symbol>@markPrice@1s, !markPrice@arr, 和 !markPrice@arr@1s 返回内容新增字段 P 表示估计结算价
- 新增行情连续合约K线推送 <pair>_<contractType>@continuousKline_<interval>

REST API

- 接口 GET /fapi/v1/premiumIndex 返回内容新增字段 "estimatedSettlePrice" 表示估计结算价。
- 接口 GET /fapi/v1/exchangeInfo 返回内容新增字段:
 - "pair" 标的交易对
 - "contractType" 合约类型
 - "deliveryDate" 交割日期
 - "onboardDate" 上线日期
- 新增接口 GET /fapi/v1/continuousKlines 获取连续合约K线数据

ENUM

- 合约类型:
 - PERPETUAL 永续合约
 - CURRENT_MONTH 当月交割合约
 - NEXT_MONTH 次月交割合约
 - CURRENT_QUARTER 当季交割合约
 - NEXT_QUARTER 次季交割合约

2020-11-27

- 新增接口 `GET /fapi/v1/commissionRate` 以查询用户交易手续费率。

2020-11-13

WEB SOCKET STREAM

- 为了给用户提供更安全稳定的服务，`<symbol>depth@0ms` and `<symbol>@depth<level>@0ms` 的更新频率调整为根据数据流量总量和其他客观情况动态调整

2020-11-10

- 接口 `GET /fapi/v1/exchangeInfo` 新增返回字段 "marginAsset" 表示保证金资产
- 接口 `GET /fapi/v2/account` 新增返回字段 "positionAmt" 表示持仓数量

2020-11-09

WEB SOCKET USER DATA STREAM

USER-DATA-STREAM 中的事件 `ACCOUNT_UPDATE` 推送规则作出了以下更新和优化：

- 当用户某项资产发生变化时：
 - 资产项目 "B" 中仅会推送本次发生变化的资产及其余额
 - 其他资产不会被推送，即便资产不为0
 - 如果资产变化不涉及持仓变化，持仓项目 "P" 将仅返回空 []
- 当合约某symbol的持仓或全逐仓配置发生变动时
 - "P" 中会推送该symbol对应的 "BOTH" 方向上的持仓详情
 - 如果是多空方向上发生持仓变动，"P" 中会推送该symbol发生持仓变动的对应 "LONG" 或 "SHORT" 方向上的持仓详情
 - 该symbol 上被初始化过的 "LONG" 或 "SHORT" 方向的逐仓持仓，也会被推送
 - 所以该symbol 上推送的 position 方向组合，由具体场景决定()
 - 其他symbol 的所有持仓信息都不会被推送，即使其持仓不为0
- 简言之，您应该通过相关的rest接口 (`GET /fapi/v2/account` 和 `GET /fapi/v2/positionRisk`) 获取资产和头寸的全量信息；通过 Websocket USER-DATA-STREAM 中的事件 `ACCOUNT_UPDATE` 对本地缓存的资产或头寸数据进行增量更新。
- 可以访问[这里](#) 获取示例以帮助对本次优化升级的理解

2020-10-27

WEB SOCKET STREAM

- 单个连接可订阅的最大stream数量调整为 200

2020-10-10

WEBSOCKET

- 新增 WebSocket 综合指数交易对信息更新 `<symbol>@compositeIndex`。

2020-10-09

- 新增接口 `GET /fapi/v1/indexInfo` 以获取交易对为综合指数的基础成分信息。

2020-09-18

- 新增 API 交易量化规则指标查询接口 `GET /fapi/v1/apiTradingStatus`。

2020-09-16

- 新增杠杆代币历史净值K线接口 `GET /fapi/v1/lvtKlines`。
杠杆代币净值系统基于合约架构，故该接口采用fapi。

WEBSOCKET

- 新增 WebSocket 杠杆代币信息更新 `<tokenName>@tokenNav` 和
净值K线更新 `<tokenName>@nav_Kline_<interval>`。
杠杆代币净值系统基于合约架构，故该推送采用合约WS服务。

2020-09-09

- 一些过期或者被取消的订单将在未来开始逐步不会从API的接口返回。
 - 被移除的订单需要满足如下条件：
 - 订单的最终状态为 `CANCELED` 或者 `EXPIRED` 并且
 - 订单没有任何的成交记录, 并且
 - 订单生成时间 + 7天 < 当前时间
 - 如下的接口会受影响:
 - `GET /fapi/v1/order`
 - `GET /fapi/v1/allOrders`

2020-08-14

- 接口 `GET /fapi/v1/premiumIndex` 新增返回字段 "indexPrice", 表示现货指数价格。
- 以下websocket 行情，新增返回字段 "i" 表示现货指数价格：
 - `<symbol>@markPrice`,
 - `<symbol>@markPrice@1s`,
 - `!markPrice@arr`,
 - `!markPrice@arr@1s`

2020-08-12

- 新增接口 `GET /fapi/v1/forceOrders` 以获取用户强平订单历史.

2020-07-30

- 新增接口 `GET /fapi/v1/adlQuantile` 以获取持仓ADL队列位置估算分数

2020-07-17

- 接口 `GET /fapi/v1/income` 权重调整为 20

2020-07-02

WEBSOCKET

- "ACCOUNT_UPDATE" 事件新增返回字段 "m" 表示事件推出缘由。
- "ORDER_TRADE_UPDATE" 事件新增返回字段 "rp" 表示该交易实现损益。

2020-06-15

- 接口 `GET /fapi/v2/account`, `GET /fapi/v2/balance` 返回内容新增字段:
 - `availableBalance`
 - `maxWithdrawAmount`

2020-06-04

- 新增 `/fapi/v2/` 接口, 较v1对应接口性能有较大提升:
 - `GET /fapi/v2/account`
 - `GET /fapi/v2/balance`

2020-06-02

- 新增 `/fapi/v2/` 接口 `GET /fapi/v2/positionRisk`:
 - 允许用户指定symbol查询
 - 市场上所有symbol都可以被查询
 - 返回内容有效区分单向持仓模式和双向持仓模式
 - 较 '`/fapi/v1/positionRisk`' 性能有较大改善

2020-05-18

- 新增参数 `closePosition` 于下单接口 `POST /fapi/v1/order`, 表示条件全部平仓:
如果一个`STOP_MARKET` 或 `TAKE_PROFIT_MARKET` 条简单设置了 `closePosition=true` 并被触发了, 当时持有所有多头仓位(若为卖单)或当时持有所有空头仓位(若为买单)将会被平仓。
- 新增返回字段 `closePosition` 于以下接口表示是否为条件全平仓单:
 - `POST /fapi/v1/order`
 - `POST /fapi/v1/batchOrders`
 - `GET /fapi/v1/order`
 - `DELETE /fapi/v1/order`
 - `DELETE /fapi/v1/batchOrders`
 - `GET /fapi/v1/openOrder`
 - `GET /fapi/v1/openOrders`
 - `GET /fapi/v1/allOrders`

2020-05-18

- 一些过期或者被取消的订单将在未来开始逐步不会从API的接口返回, 但是还可以从网页端查询到。
 - 被移除的订单需要满足如下条件:
 - 订单的最终状态为 `CANCELED` 或者 `EXPIRED`, 并且
 - 订单没有任何的成交记录, 并且
 - 订单生成时间 + 30天 < 当前时间
 - 如下的接口会受影响:
 - `GET /fapi/v1/order`
 - `GET /fapi/v1/allOrders`

2020-05-15

- Wesocket 行情消息 `<symbol>@bookTicker` 和 `!bookTicker` 增加返回字段:
 - `E` 表示事件推出事件
 - `T` 表示撮合时间

2020-05-14

- 以下接口返回内容增加 `time` 字段, 表示撮合引擎时间:
 - `GET /fapi/v1/ticker/price`
 - `GET /fapi/v1/ticker/bookTicker`
 - `GET /fapi/v1/openInterest`

2020-05-11

- 新增接口 `POST /fapi/v1/countdownCancelAll` 以实现倒计时自动撤单。
 - 该接口可以被用于确保在倒计时结束时撤销指定symbol上的所有挂单。
 - 在使用这个功能时，接口应像心跳一样在倒计时内被反复调用，以便可以取消既有的倒计时并开始新的倒数计时设置。

2020-05-06

REST 接口

- 接口 `GET /fapi/v1/leverageBracket` 调整为 USER-DATA 权限访问，需要验签以及timestamp

WEBSOCKET 账户信息推送

- 请注意：当某一持仓发生“FUNDING FEE”时，事件 `ACCOUNT_UPDATE` 将只会推送相关的用户资产余额信息和持仓信息，而不会推送其余无关的资产和持仓信息。
 - 当用户某全仓持仓发生“FUNDING FEE”时，事件 `ACCOUNT_UPDATE` 将只会推送相关的用户资产余额信息 (B) (仅推送 FUNDING FEE 发生相关的资产余额信息)，而不会推送任何持仓信息 (P)。
 - 当用户某逐仓持仓发生“FUNDING FEE”时，事件 `ACCOUNT_UPDATE` 将只会推送相关的用户资产余额信息 (B) (仅推送“FUNDING FEE”所使用的资产余额信息)，和相关的持仓信息 (P) (仅推送这笔“FUNDING FEE”发生所在的持仓信息)，其余持仓信息不会被推送。

2020-04-25

- 用户“订单/交易更新推送” `ORDER_TRADE_UPDATE` 新增以下字段：
 - `cp` 表示是否为平仓条件单
 - `AP` 表示追踪止损单的追踪止损激活价格
 - `cr` 表示追踪止损单的追踪止损回调比例
- 新增账户信息推送事件：“追加保证金通知” `MARGIN_CALL`。

2020-04-17

- 下单接口支持新的可选参数 `newOrderRespType` 表示下单响应类型。支持 `ACK` 和 `RESULT`，如果 `newOrderRespType= RESULT`：
 - `MARKET` 订单将直接返回成交(FILLED)结果；
 - 配合使用特殊 `timeInForce` 的 `LIMIT` 订单将直接返回成交/过期(FILLED/EXPIRED)结果。

2020-04-14

WEB SOCKET 连接限制

- Websocket服务器每秒最多接受10个消息。消息包括：
 - PING帧
 - PONG帧
 - JSON格式的消息，比如订阅，断开订阅。
- 如果用户发送的消息超过限制，连接会被断开连接。反复被断开连接的IP有可能被服务器屏蔽。
- 单个连接最多可以订阅 200 个Streams。

2020-04-09

- 新增接口合约大数据 `GET /futures/data/takerlongshortRatio` 以查询合约主动买卖量

2020-04-08

- 新增接口 `GET /fapi/v1/positionSide/dual` 以查询用户当前持仓模式

- 新增接口 `POST /fapi/v1/batchOrders` 以实现批量下单

2020-04-06

- 请注意 账户信息推送 事件 "Balance和Position更新推送"(ACCOUNT_UPDATE)将不再未发生更新时推送，具体规则如下：
 - 仅当账户信息有变动时(包括资金、仓位、保证金模式等发生变化)，才会推送此事件；
 - 订单状态变化没有引起账户和持仓变化的，不会推送此事件；
 - 每次推送的position 信息，仅包含当前持仓不为0或逐仓仓位保证金不为0的symbol position。
- 新增接口 `POST /fapi/v1/positionSide/dual` 更改持仓模式：双向或单向持仓模式。
- 以下接口新增参数 `positionSide` 用以支持单向/双向持仓模式，表示持仓方向：
 - `POST /fapi/v1/order`
 - `POST /fapi/v1/positionMargin`
- 以下接口新增返回字段 `positionSide` 用以支持单向/双向持仓模式，表示持仓方向：
 - `POST /fapi/v1/order`
 - `GET /fapi/v1/order`
 - `DELETE /fapi/v1/order`
 - `DELETE /fapi/v1/batchOrders`
 - `GET /fapi/v1/openOrder`
 - `GET /fapi/v1/openOrders`
 - `GET /fapi/v1/allOrders`
 - `GET /fapi/v1/account`
 - `POST /fapi/v1/positionMargin`
 - `GET /fapi/v1/positionMargin/history`
 - `GET /fapi/v1/positionRisk`
 - `GET /fapi/v1/userTrades`
- 账户信息推送 事件 "Balance和Position更新推送"(ACCOUNT_UPDATE)和 "订单/交易更新推送"(ORDER_TRADE_UPDATE)中新增字段 `ps` 表示持仓方向。

2020-03-30

- 新增接口合约大数据：
 - `GET /futures/data/openInterestHist`
 - `GET /futures/data/topLongShortAccountRatio`
 - `GET /futures/data/topLongShortPositionRatio`
 - `GET /futures/data/globalLongShortAccountRatio`

2020-02-26

- 新增订单类型:跟踪止损 `TRAILING_STOP_MARKET`

2020-02-20

- 新增接口以查询指定的当前挂单: `GET /fapi/v1/openOrder`

2020-02-17

- `<symbol>@ticker` 与 `!ticker@arr` 更新频率提升为1000ms
- 新增500ms更新的增量深度信息流选项: `<symbol>@depth@500ms`
- 新增500ms更新的有限档深度信息流选项: `<symbol>@depth<level>@500ms`

2020-02-12

- Java SDK和代码示例 发布
 - 实现每秒更新的标记价格信息流选项:
`<symbol>@markPrice@1s` and `!markPrice@arr@1s`
-

2020-02-05

- 新增接口`GET /fapi/v1/leverageBracket`: 查询杠杆分层标准。
-

2020-01-19

- "cumQty" 字段将于未来几周从 `DELETE /fapi/v1/order` , `DELETE /fapi/v1/batchOrders` 等 `order` 相关接口的返回内容中去除，请使用 "executedQty" 字段予以替代。
-

2020-1-17

- Python SDK和代码示例 发布
-

2020-1-6

- 实现实时更新的增量深度信息流选项: `<symbol>@depth@0ms`
-

2020-1-3

- 新增逐仓相关接口：
 - `POST /fapi/v1/marginType`
 - `POST /fapi/v1/positionMargin`
 - `GET /fapi/v1/positionMargin/history`
 - 接口`GET /fapi/v1/positionRisk`新增返回内容:
 - `marginType`
 - `isolatedMargin`
 - 接口`GET /fapi/v1/account`新增返回内容 : `isolated`
 - `ACCOUNT_UPDATE` Balance和Position更新推送 增加 :
 - "cw": 除去逐仓保证金的钱包余额
 - "mt": 保证金模式
 - "iw": 若为逐仓，仓位保证金
-

2019-12-19

- 新增接口获取市场当前未平仓合约数 : `GET /fapi/v1/openInterest`
-

2019-12-18

- 新增账户信息推送事件 : `listenKeyExpired`。
-

2019-12-12

- 新增接口撤销指定symbol的所有订单: `DELETE /fapi/v1/allOpenOrders`
 - 新增接口批量撤销订单 : `DELETE /fapi/v1/batchOrders`
 - 新增支持仅减仓`reduceOnly`的订单类型 :
 - `TAKE_PROFIT`
 - `TAKE_PROFIT_MARKET`
 - `STOP`
 - `STOP_MARKET`
-

2019-11-29

- 新增接口获取市场强平订单 : `GET /fapi/v1/allForceOrders`
 - 新增市场行情推送 :
 - 强平订单 : `<symbol>@forceOrder`
 - 全市场强平订单 : `!forceOrder@arr`
-

2019-11-25

- `GET /fapi/v1/account` 新增返回内容: `positions`
 - 以下接口新增返回值 `time` 表示订单创建时间:
 - `GET /fapi/v1/openOrders`
 - `GET /fapi/v1/order`
 - `GET /fapi/v1/allOrders`
-

2019-11-15

- Websocket 新增市场行情流 :
 - `!miniTicker@arr`: 全市场的精简Ticker更新
 - `!ticker@arr`: 全市场的完整Ticker更新
-

2019-11-12

- WSS 支持实时订阅和取消数据流。
-

2019-11-05

- 新增订单类型:
 - `STOP_MARKET`止损市价单 ,
 - `TAKE_PROFIT_MARKET`止盈市价单
 - 下单新增可选参数: `workingType` 可选`stopPrice`由 "CONTRACT_PRICE" 或 "MARK_PRICE" 触发
 - USER-DATA-STREAMS新增:
 - `ORDER_TRADE_UPDATE`订单/交易 更新推送 增加 :
 - "T": 撮合时间
 - "wt": workingType
 - `ACCOUNT_UPDATE` Balance和Position更新推送 增加 : "T": 撮合时间
-

2019-10-28

- 新增接口查询账户损益资金流水 : `GET /fapi/v1/income`
-

2019-10-25

- 账户信息推送事件 `ACCOUNT_UPDATE` 增加字段 "up" , 表示持仓未实现盈亏。
 - 账户信息推送事件 `ORDER_TRADE_UPDATE` 增加字段 "R" , 表示该成交是否作为只减仓单。
-

2019-10-24

- 新增最优挂单信息行情流: `<symbol>@bookTicker` 与 `!bookTicker`
 - 新增有限档深度信息行情流 : `<symbol>@depth<levels>` 与 `<symbol>@depth<levels>@100ms`
 - 更新频率达到100ms的更快的增量深度信息流选项: `<symbol>@depth@100ms`
 - `Websocket` 行情推送 增加 `Update Speed` 更新速度
-

2019-10-18

- 新增接口 `POST /fapi/v1/leverage` 以调整开仓杠杆倍数。
 - 接口 `GET /fapi/v1/positionRisk` 的返回内容中新增字段 :
 - `"leverage"`: 当前开仓杠杆倍数 ;
 - `"maxNotionalValue"`: 当前开仓杠杆倍数下的名义价值上限。
 - `MARKET` 市价单支持 `reduceOnly` 只减仓参数。
-

2019-10-14

- 新增接口 `GET /fapi/v1/fundingRate`: 获取资金费率历史。
-

2019-10-11

- 账户信息推送事件 `ORDER_TRADE_UPDATE` 增加字段 `"m"` , 表示该成交是否作为挂单成交
-

2019-10-08

- 新增限价指令订单参数 `reduceOnly` : 只减仓
 - 新增订单类型 `TAKE_PROFIT` : 止盈单
-

2019-09-20

- `GET /fapi/v1/account` 新增返回值:
`maxWithdrawAmount`, `openOrderInitialMargin`, `positionInitialMargin`
 - `GET /fapi/v1/positionRisk` 新增返回值:
`liquidationPrice`
-

基本信息

SDK和代码示例

免责声明:

- 以下SDK由合作方和用户提供，**非官方制作**行为。仅做熟悉api接口和学习使用，请广大用户谨慎使用并根据自身情况自行拓展研发。
- Binance 官方不对SDK的安全和性能做任何承诺，亦不会对使用SDK引起的风险甚至损失承担责任。

PYTHON3

SDK 1: 可以通过以下方式获取Binance Futures Connector SDK :

- 访问 <https://github.com/Binance-docs/binance-futures-connector-python>

- 执行以下命令：

```
pip install binance-futures-connector
```

SDK 2: 可以通过以下方式获取SDK :

- 访问 https://github.com/Binance-docs/Binance_Futures_python

- 执行以下命令：

```
git clone https://github.com/Binance-docs/Binance_Futures_python.git
```

JAVA

可以通过以下方式获取SDK :

* 访问 https://github.com/Binance-docs/Binance_Futures_Java,

- 执行以下命令：

```
git clone https://github.com/Binance-docs/Binance_Futures_Java.git
```

Rest 基本信息

- 接口可能需要用户的 API Key , 如何创建API-KEY请参考[这里](#)
- 本篇列出REST接口的baseurl **<https://fapi.binance.com>**
- 所有接口的响应都是JSON格式
- 响应中如有数组 , 数组元素以时间升序排列 , 越早的数据越提前。
- 所有时间、时间戳均为UNIX时间 , 单位为毫秒
- 所有数据类型采用JAVA的数据类型定义

testnet

- 本篇接口亦可接入testnet测试平台使用
- **testnet**的 REST baseurl 为 "<https://testnet.binancefuture.com>"
- **testnet**的 Websocket baseurl 为 "<wss://stream.binancefuture.com>"

HTTP 返回代码

- HTTP **4XX** 错误码用于指示错误的请求内容、行为、格式。
- HTTP **403** 错误码表示违反WAF限制(Web应用程序防火墙)。
- HTTP **429** 错误码表示警告访问频次超限 , 即将被封IP
- HTTP **418** 表示收到429后继续访问 , 于是被封了。
- HTTP **5XX** 错误码用于指示Binance服务侧的问题。
 - 如果返回内容里包含了报错信息 "**Request occur unknown error.**" , 请稍后重试请求。
- HTTP **503** 表示三种可能 :
 - 如果返回内容里包含了报错信息 "**Unknown error, please check your request or try again later.**" , 则表示API服务端已经向业务核心提交了请求但未能获取响应 , 特别需要注意的是其不代表请求失败 , 而是未知。很可能已经得到了执行 , 也有可能执行失败 , 需要做进一步确认。
 - 如果返回内容里包含了报错信息 "**Service Unavailable.**" , 则表示本次API请求失败。这种情况下可能是服务暂不可用 , 您需要稍后重试。
 - 如果返回内容里包含了报错信息 "**Internal error; unable to process your request. Please try again.**" , 则表示本次API请求失败。这种情况下您如果需要的话可以选择立即重试。

接口错误代码

- 每个接口都有可能抛出异常

异常响应格式如下 :

```
{
  "code": -1121,
  "msg": "Invalid symbol."
}
```

- 具体的错误码及其解释在错误代码

接口的基本信息

- `GET`方法的接口，参数必须在`query string`中发送。
- `POST`, `PUT`, 和 `DELETE` 方法的接口，参数可以在`query string`中发送，也可以在`request body`中发送(content type `application/x-www-form-urlencoded`)。允许混合这两种方式发送参数。但如果同一个参数名在`query string`和`request body`中都有，`query string`中的会被优先采用。
- 对参数的顺序不做要求。

访问限制

- 在`/fapi/v1/exchangeInfo`接口中`rateLimits`数组里包含有REST接口(不限于本篇的REST接口)的访问限制。包括带权重的访问频次限制、下单速率限制。本篇枚举定义章节有限制类型进一步说明。
- 违反上述任何一个访问限制都会收到HTTP 429，这是一个警告。

① 请注意，若用户被认定利用频繁挂撤单且故意低效交易意图发起攻击行为，Binance有权视具体情况进一步加强对其访问限制。

IP 访问限制

- 每个请求将包含一个`X-MBX-USED-WEIGHT-(intervalNum)(intervalLetter)`的头，其中包含当前IP所有请求的已使用权重。
- 每个路由都有一个“权重”，该权重确定每个接口计数的请求数。较重的接口和对多个交易对进行操作的接口将具有较重的“权重”。
- 收到429时，您有责任作为API退回而不向其发送更多的请求。
- 如果屡次违反速率限制和/或在收到429后未能退回，将导致API的IP被禁(http状态418)。**
- 频繁违反限制，封禁时间会逐渐延长，对于重复违反者，将会被封从2分钟到3天。
- 访问限制是基于IP的，而不是API Key**

① 强烈建议您尽可能多地使用websocket消息获取相应数据，既可以保障消息的及时性，也可以减少请求带来的访问限制压力。

下单频率限制

- 每个下单请求回报将包含一个`X-MBX-ORDER-COUNT-(intervalNum)(intervalLetter)`的头，其中包含当前账户已用的下单限制数量。
- 被拒绝或不成功的下单并不保证回报中包含以上头内容。
- 下单频率限制是基于每个账户计数的。**

接口鉴权类型

- 每个接口都有自己的鉴权类型，鉴权类型决定了访问时应当进行何种鉴权
- 如果需要 API-key，应当在HTTP头中以`X-MBX-APIKEY`字段传递
- API-key 与 API-secret 是大小写敏感的
- 可以在网页用户中心修改API-key 所具有的权限，例如读取账户信息、发送交易指令、发送提现指令

鉴权类型	描述
------	----

鉴权类型	描述
NONE	不需要鉴权的接口
TRADE	需要有效的API-KEY和签名
USER_DATA	需要有效的API-KEY和签名
USER_STREAM	需要有效的API-KEY
MARKET_DATA	需要有效的API-KEY

需要签名的接口 (TRADE 与 USER_DATA)

- 调用这些接口时，除了接口本身所需的参数外，还需要传递 `signature` 即签名参数。
- 签名使用 `HMAC SHA256` 算法。API-KEY所对应的API-Secret作为 `HMAC SHA256` 的密钥，其他所有参数作为 `HMAC SHA256` 的操作对象，得到的输出即为签名。
- 签名大小写不敏感。
- 当同时使用query string和request body时，`HMAC SHA256`的输入query string在前，request body在后

时间同步安全

- 签名接口均需要传递 `timestamp` 参数，其值应当是请求发送时刻的unix时间戳(毫秒)
- 服务器收到请求时会判断请求中的时间戳，如果是5000毫秒之前发出的，则请求会被认为无效。这个时间窗口值可以通过发送可选参数 `recvWindow` 来自定义。
- 另外，如果服务器计算得出客户端时间戳在服务器时间的‘未来’一秒以上，也会拒绝请求。

逻辑伪代码：

```
if (timestamp < (serverTime + 1000) && (serverTime - timestamp) <= recvWindow) {
    // process request
} else {
    // reject request
}
```

关于交易时效性 互联网状况并不100%可靠，不可完全依赖，因此你的程序本地到币安服务器的时延会有抖动。这是我们设置 `recvWindow` 的目的所在，如果你从事高频交易，对交易时效性有较高的要求，可以灵活设置 `recvWindow` 以达到你的要求。

① 不推荐使用5秒以上的recvWindow

POST /FAPI/V1/ORDER 的示例

以下是在linux bash环境下使用 `echo openssl` 和 `curl` 工具实现的一个调用接口下单的示例 `apikey`、`secret` 仅供示范

Key	Value
apiKey	dbefbc809e3e83c283a984c3a1459732ea7db1360ca80c5c2c8867408d28cc83
secretKey	2b5eb11e18796d12d88f13dc27dbbd02c2cc51ff7059765ed9821957d82bb4d9
参数	取值
symbol	BTCUSDT

参数	取值
side	BUY
type	LIMIT
timeInForce	GTC
quantity	1
price	9000
recvWindow	5000
timestamp	1591702613943

示例 1: 所有参数通过 QUERY STRING 发送**示例1:****HMAC SHA256 签名:**

```
$ echo -n "symbol=BTCUSDT&side=BUY&type=LIMIT&quantity=1&price=9000&timeInForce=GTC&recvWindow=5000&timestamp=1591702613943" | openssl dgst -sha256 -sign /path/to/privkey.pem -out signature.txt
(stdin)= 3c661234138461fcc7a7d8746c6558c9842d4e10870d2ecbedf7777cad694af9
```

curl 调用:

```
(HMAC SHA256)
$ curl -H "X-MBX-APIKEY: dbefbc809e3e83c283a984c3a1459732ea7db1360ca80c5c2c8867408d28cc83" -X POST 'https://api.binance.com/api/v3/order?symbol=BTCUSDT&side=BUY&type=LIMIT&quantity=1&price=9000&timeInForce=GTC&recvWindow=5000&timestamp=1499827319559'
```

• queryString:

```
symbol=BTCUSDT
&side=BUY
&type=LIMIT
&timeInForce=GTC
&quantity=1
&price=0.1
&recvWindow=5000
&timestamp=1499827319559
```

示例 2: 所有参数通过 REQUEST BODY 发送**示例2:****HMAC SHA256 签名:**

```
$ echo -n "symbol=BTCUSDT&side=BUY&type=LIMIT&quantity=1&price=9000&timeInForce=GTC&recvWindow=5000&timestamp=1591702613943" | openssl dgst -sha256 -sign /path/to/privkey.pem -out signature.txt
(stdin)= 3c661234138461fcc7a7d8746c6558c9842d4e10870d2ecbedf7777cad694af9
```

curl 调用:

```
(HMAC SHA256)
$ curl -H "X-MBX-APIKEY: dbefbc809e3e83c283a984c3a1459732ea7db1360ca80c5c2c8867408d28cc83" -X POST 'https://api.binance.com/api/v3/order?symbol=BTCUSDT&side=BUY&type=LIMIT&quantity=1&price=9000&timeInForce=GTC&recvWindow=5000&timestamp=1499827319559'
```

- **requestBody:**

```
symbol=BTCUSDT
&side=BUY
&type=LIMIT
&timeInForce=GTC
&quantity=1
&price=9000
&recvWindow=5000
&timestamp=1591702613943
```

示例 3: 混合使用 QUERY STRING 与 REQUEST BODY

示例3:

HMAC SHA256 签名:

```
$ echo -n "symbol=BTCUSDT&side=BUY&type=LIMIT&quantity=1&price=9000&timeInForce=GTC&recvWindow=5000&timestamp=1591702613943" | openssl dgst -sha256 -hmac $APIKEY
(stdin)= 3c661234138461fcc7a7d8746c6558c9842d4e10870d2ecbedf7777cad694af9
```

curl 调用:

```
(HMAC SHA256)
$ curl -H "X-MBX-APIKEY: dbefbc809e3e83c283a984c3a1459732ea7db1360ca80c5c2c8867408d28cc83" -X POST 'https://api.binance.com/api/v3/order'
```

- **queryString:** symbol=BTCUSDT&side=BUY&type=LIMIT&timeInForce=GTC
- **requestBody:** quantity=1&price=9000&recvWindow=5000×tamp=1591702613943

请注意，示例3中的签名有些许不同，在"GTC"和"quantity=1"之间没有"&"字符。

公开API参数

术语解释

- **base asset** 指一个交易对的交易对象，即写在靠前部分的资产名
- **quote asset** 指一个交易对的定价资产，即写在靠后部分资产名

枚举定义

交易对类型:

- FUTURE 期货

合约类型 (contractType):

- PERPETUAL 永续合约
- CURRENT_MONTH 当月交割合约
- NEXT_MONTH 次月交割合约
- CURRENT_QUARTER 当季交割合约
- NEXT_QUARTER 次季交割合约
- PERPETUAL_DELIVERING 交割结算中合约

合约状态 (contractStatus, status):

- PENDING_TRADING 待上市

- TRADING 交易中
- PRE_DELIVERING 预交割
- DELIVERING 交割中
- DELIVERED 已交割
- PRE_SETTLE 预结算
- SETTLING 结算中
- CLOSE 已下架

订单状态 (status):

- NEW 新建订单
- PARTIALLY_FILLED 部分成交
- FILLED 全部成交
- CANCELED 已撤销
- REJECTED 订单被拒绝
- EXPIRED 订单过期(根据timeInForce参数规则)

订单种类 (orderTypes, type):

- LIMIT 限价单
- MARKET 市价单
- STOP 止损限价单
- STOP_MARKET 止损市价单
- TAKE_PROFIT 止盈限价单
- TAKE_PROFIT_MARKET 止盈市价单
- TRAILING_STOP_MARKET 跟踪止损单

订单方向 (side):

- BUY 买入
- SELL 卖出

持仓方向:

- BOTH 单一持仓方向
- LONG 多头(双向持仓下)
- SHORT 空头(双向持仓下)

有效方式 (timeInForce):

- GTC - Good Till Cancel 成交为止
- IOC - Immediate or Cancel 无法立即成交(吃单)的部分就撤销
- FOK - Fill or Kill 无法全部立即成交就撤销
- GTX - Good Till Crossing 无法成为挂单方就撤销

条件价格触发类型 (workingType)

- MARK_PRICE
- CONTRACT_PRICE

响应类型 (newOrderRespType)

- ACK
- RESULT

K线间隔:

m -> 分钟; h -> 小时; d -> 天; w -> 周; M -> 月

- 1m
- 3m
- 5m
- 15m
- 30m
- 1h
- 2h
- 4h
- 6h

- 8h
- 12h
- 1d
- 3d
- 1w
- 1M

限制种类 (rateLimitType)

REQUEST_WEIGHT

```
{
  "rateLimitType": "REQUEST_WEIGHT",
  "interval": "MINUTE",
  "intervalNum": 1,
  "limit": 2400
}
```

ORDERS

```
{
  "rateLimitType": "ORDERS",
  "interval": "MINUTE",
  "intervalNum": 1,
  "limit": 1200
}
```

- REQUESTS_WEIGHT 单位时间请求权重之和上限
- ORDERS 单位时间下单(撤单)次数上限

限制间隔

- MINUTE

过滤器

过滤器，即Filter，定义了一系列交易规则。共有两类，分别是针对交易对的过滤器symbol filters，和针对整个交易所的过滤器exchange filters(暂不支持)

交易对过滤器

PRICE_FILTER 价格过滤器

/exchangeInfo 响应中的格式：

```
{
  "filterType": "PRICE_FILTER",
  "minPrice": "0.0000100",
  "maxPrice": "100000.0000000",
  "tickSize": "0.0000100"
}
```

价格过滤器用于检测order订单中price参数的合法性

- minPrice 定义了 price/stopPrice 允许的最小值

- `maxPrice` 定义了 `price`/`stopPrice` 允许的最大值。
- `tickSize` 定义了 `price`/`stopPrice` 的步进间隔，即 `price` 必须等于 `minPrice + (tickSize 的整数倍)` 以上每一项均可为 0，为 0 时代表这一项不再做限制。

逻辑伪代码如下：

- `price >= minPrice`
- `price <= maxPrice`
- `(price - minPrice) % tickSize == 0`

LOT_SIZE 订单尺寸

*/exchangeInfo 响应中的格式**

```
{
  "filterType": "LOT_SIZE",
  "minQty": "0.00100000",
  "maxQty": "100000.00000000",
  "stepSize": "0.00100000"
}
```

lots 是拍卖术语，这个过滤器对订单中的 `quantity` 也就是数量参数进行合法性检查。包含三个部分：

- `minQty` 表示 `quantity` 允许的最小值。
- `maxQty` 表示 `quantity` 允许的最大值。
- `stepSize` 表示 `quantity` 允许的步进值。

逻辑伪代码如下：

- `quantity >= minQty`
- `quantity <= maxQty`
- `(quantity - minQty) % stepSize == 0`

MARKET_LOT_SIZE 市价订单尺寸

参考 `LOT_SIZE`，区别仅在于对市价单还是限价单生效

MAX_NUM_ORDERS 最多订单数

/exchangeInfo 响应中的格式:

```
{
  "filterType": "MAX_NUM_ORDERS",
  "limit": 200
}
```

定义了某个交易对最多允许的挂单数量(不包括已关闭的订单)

普通订单与条件订单均计算在内

MAX_NUM_ALGO_ORDERS 最多条件订单数

/exchangeInfo format:

```
{
  "filterType": "MAX_NUM_ALGO_ORDERS",
  "limit": 100
}
```

定义了某个交易对最多允许的条件订单的挂单数量(不包括已关闭的订单)。

条件订单目前包括 `STOP`, `STOP_MARKET`, `TAKE_PROFIT`, `TAKE_PROFIT_MARKET`, 和 `TRAILING_STOP_MARKET`

PERCENT_PRICE 价格振幅过滤器

`/exchangeInfo` 响应中的格式:

```
{
  "filterType": "PERCENT_PRICE",
  "multiplierUp": "1.1500",
  "multiplierDown": "0.8500",
  "multiplierDecimal": 4
}
```

`PERCENT_PRICE` 定义了基于标记价格计算的挂单价格的可接受区间。

挂单价格必须同时满足以下条件 :

- 买单: `price` <= `markPrice` * `multiplierUp`
- 卖单: `price` >= `markPrice` * `multiplierDown`

MIN_NOTIONAL 最小名义价值

`/exchangeInfo` 响应中的格式:

```
{
  "filterType": "MIN_NOTIONAL",
  "notional": "1"
}
```

`MIN_NOTIONAL` 过滤器定义了交易对订单所允许的最小名义价值(成交额)。 订单的名义价值是 `价格 * 数量`。 由于 `MARKET` 订单没有价格，因此会使用 mark price 计算。

Postman Collections

现在你可以通过 `Postman collection` 来快速体验、使用 API 接口。

如果想了解更多如果使用 Postman，请访问 Binance API Postman

行情接口

测试服务器连通性 PING

`GET /fapi/v1/ping`

响应:

```
[]
```

测试能否联通

权重: 1**参数:** NONE

获取服务器时间

响应:

```
{  
    "serverTime": 1499827319559 // 当前的系统时间  
}
```

GET /fapi/v1/time

获取服务器时间

权重: 1**参数:** NONE

获取交易规则和交易对

响应:

```
{  
    "exchangeFilters": [],  
    "rateLimits": [ // API访问的限制  
        {  
            "interval": "MINUTE", // 按照分钟计算  
            "intervalNum": 1, // 按照1分钟计算  
            "limit": 2400, // 上限次数  
            "rateLimitType": "REQUEST_WEIGHT" // 按照访问权重来计算  
        },  
        {  
            "interval": "MINUTE",  
            "intervalNum": 1,  
            "limit": 1200,  
            "rateLimitType": "ORDERS" // 按照订单数量来计算  
        }  
    "serverTime": 1565613908500, // 请忽略。如果需要获取当前系统时间, 请查询接口 “GET /fapi/v1/time”  
    "assets": [ // 资产信息  
        {  
            "asset": "BUSD",  
            "marginAvailable": true, // 是否可用作保证金  
            "autoAssetExchange": 0 // 保证金资产自动兑换阈值  
        },  
        {  
            "asset": "USDT",  
            "marginAvailable": true, // 是否可用作保证金  
            "autoAssetExchange": 0 // 保证金资产自动兑换阈值  
        }  
    ]  
}
```

```

    },
    {
        "asset": "BNB",
        "marginAvailable": false, // 是否可用作保证金
        "autoAssetExchange": null // 保证金资产自动兑换阈值
    }
],
"symbols": [ // 交易对信息
{
    "symbol": "BLZUSDT", // 交易对
    "pair": "BLZUSDT", // 标的交易对
    "contractType": "PERPETUAL", // 合约类型
    "deliveryDate": 4133404800000, // 交割日期
    "onboardDate": 1598252400000, // 上线日期
    "status": "TRADING", // 交易对状态
    "maintMarginPercent": "2.5000", // 请忽略
    "requiredMarginPercent": "5.0000", // 请忽略
    "baseAsset": "BLZ", // 标的资产
    "quoteAsset": "USDT", // 报价资产
    "marginAsset": "USDT", // 保证金资产
    "pricePrecision": 5, // 价格小数点位数(仅作为系统精度使用, 注意同tickSize 区分)
    "quantityPrecision": 0, // 数量小数点位数(仅作为系统精度使用, 注意同stepSize 区分)
    "baseAssetPrecision": 8, // 标的资产精度
    "quotePrecision": 8, // 报价资产精度
    "underlyingType": "COIN",
    "underlyingSubType": ["STORAGE"],
    "settlePlan": 0,
    "triggerProtect": "0.15", // 开启"priceProtect"的条件订单的触发阈值
    "filters": [
        {
            "filterType": "PRICE_FILTER", // 价格限制
            "maxPrice": "300", // 价格上限, 最大价格
            "minPrice": "0.0001", // 价格下限, 最小价格
            "tickSize": "0.0001" // 订单最小价格间隔
        },
        {
            "filterType": "LOT_SIZE", // 数量限制
            "maxQty": "100000000", // 数量上限, 最大数量
            "minQty": "1", // 数量下限, 最小数量
            "stepSize": "1" // 订单最小数量间隔
        },
        {
            "filterType": "MARKET_LOT_SIZE", // 市价订单数量限制
            "maxQty": "590119", // 数量上限, 最大数量
            "minQty": "1", // 数量下限, 最小数量
            "stepSize": "1" // 允许的步进值
        },
        {
            "filterType": "MAX_NUM_ORDERS", // 最多订单数限制
            "limit": 200
        },
        {
            "filterType": "MAX_NUM_ALGO_ORDERS", // 最多条件订单数限制
            "limit": 100
        },
        {
            "filterType": "MIN_NOTIONAL", // 最小名义价值
            "notional": "1",
        }
    ]
}
]

```

```

    },
    {
        "filterType": "PERCENT_PRICE", // 价格比限制
        "multiplierUp": "1.1500", // 价格上限百分比
        "multiplierDown": "0.8500", // 价格下限百分比
        "multiplierDecimal": 4
    }
],
"OrderType": [ // 订单类型
    "LIMIT", // 限价单
    "MARKET", // 市价单
    "STOP", // 止损单
    "STOP_MARKET", // 止损市价单
    "TAKE_PROFIT", // 止盈单
    "TAKE_PROFIT_MARKET", // 止盈市价单
    "TRAILING_STOP_MARKET" // 跟踪止损市价单
],
"timeInForce": [ // 有效方式
    "GTC", // 成交为止，一直有效
    "IOC", // 无法立即成交(吃单)的部分就撤销
    "FOK", // 无法全部立即成交就撤销
    "GTG" // 无法成为挂单方就撤销
],
"liquidationFee": "0.010000", // 强平费率
"marketTakeBound": "0.30", // 市价吃单(相对于标记价格)允许造成最大价格偏离比例
},
],
"timezone": "UTC" // 服务器所用的时间区域
}

```

`GET /fapi/v1/exchangeInfo`

获取交易规则和交易对

权重: 1

参数: NONE

深度信息

响应:

```
{
    "lastUpdateId": 1027024,
    "E": 1589436922972, // 消息时间
    "T": 1589436922959, // 撮合引擎时间
    "bids": [ // 买单
        [
            "4.00000000", // 价格
            "431.00000000" // 数量
        ]
    ],
    "asks": [ // 卖单
        [

```

```

    "4.00000200",      // 价格
    "12.00000000"     // 数量
]
}

```

`GET /fapi/v1/depth`

权重:

limit	权重
5, 10, 20, 50	2
100	5
500	10
1000	20

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
limit	INT	NO	默认 500; 可选值:[5, 10, 20, 50, 100, 500, 1000]

近期成交

响应:

```

[
{
  "id": 28457,          // 成交ID
  "price": "4.00000100", // 成交价格
  "qty": "12.00000000", // 成交量
  "quoteQty": "48.00",   // 成交额
  "time": 1499865549590, // 时间
  "isBuyerMaker": true   // 买方是否为挂单方
}
]

```

`GET /fapi/v1/trades`

获取近期订单簿成交

权重: 5

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对

名称	类型	是否必需	描述
limit	INT	NO	默认:500，最大1000

- 仅返回订单簿成交，即不会返回保险基金和自动减仓(ADL)成交

查询历史成交(MARKET_DATA)

响应:

```
[
  [
    {
      "id": 28457,          // 成交ID
      "price": "4.00000100", // 成交价格
      "qty": "12.00000000", // 成交量
      "quoteQty": "48.00",   // 成交额
      "time": 1499865549590, // 时间
      "isBuyerMaker": true   // 买方是否为挂单方
    }
  ]
]
```

`GET /fapi/v1/historicalTrades`

查询订单簿历史成交

权重: 20

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
limit	INT	NO	默认值:500 最大值:1000.
fromId	LONG	NO	从哪一条成交id开始返回. 缺省返回最近的成交记录

- 仅返回订单簿成交，即不会返回保险基金和自动减仓(ADL)成交

近期成交(归集)

响应:

```
[
  [
    {
      "a": 26129,          // 归集成交ID
      "p": "0.01633102", // 成交价
      "q": "4.70443515", // 成交量
      "f": 27781,          // 被归集的首个成交ID
      "l": 27781,          // 被归集的末个成交ID
      "T": 1498793709153, // 成交时间
    }
  ]
]
```

```

        "m": true,           // 是否为主动卖出单
    }
]
```

GET /fapi/v1/aggTrades

归集交易与逐笔交易的区别在于，同一价格、同一方向、同一时间(100ms计算)的订单簿trade会被聚合为一条

权重: 20

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
fromId	LONG	NO	从包含fromID的成交开始返回结果
startTime	LONG	NO	从该时刻之后的成交记录开始返回结果
endTime	LONG	NO	返回该时刻为止的成交记录
limit	INT	NO	默认 500; 最大 1000.

- 如果同时发送 startTime 和 endTime，间隔必须小于一小时
- 如果没有发送任何筛选参数(fromId, startTime, endTime)，默认返回最近的成交记录
- 保险基金和自动减仓(ADL)成交不属于订单簿成交，故不会被归并聚合

K线数据

响应:

```

[
    [
        1499040000000,      // 开盘时间
        "0.01634790",       // 开盘价
        "0.80000000",       // 最高价
        "0.01575800",       // 最低价
        "0.01577100",       // 收盘价(当前K线未结束的即为最新价)
        "148976.11427815",  // 成交量
        1499644799999,      // 收盘时间
        "2434.19055334",   // 成交额
        308,                // 成交笔数
        "1756.87402397",   // 主动买入成交量
        "28.46694368",     // 主动买入成交额
        "17928899.62484339" // 请忽略该参数
    ]
]
```

GET /fapi/v1/klines

每根K线的开盘时间可视为唯一ID

权重: 取决于请求中的LIMIT参数

LIMIT参数	权重
[1,100)	1
[100, 500)	2
[500, 1000]	5
> 1000	10

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
interval	ENUM	YES	时间间隔
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	默认值:500 最大值:1500.

- 缺省返回最近的数据

连续合约K线数据

响应:

```
[
  [
    1607444700000,           // 开盘时间
    "18879.99",              // 开盘价
    "18900.00",              // 最高价
    "18878.98",              // 最低价
    "18896.13",              // 收盘价(当前K线未结束的即为最新价)
    "492.363",               // 成交量
    1607444759999,           // 收盘时间
    "9302145.66080",          // 成交额
    1874,                     // 成交笔数
    "385.983",               // 主动买入成交量
    "7292402.33267",          // 主动买入成交额
    "0"                       // 请忽略该参数
  ]
]
```

GET /fapi/v1/continuousKlines 每根K线的开盘时间可视为唯一ID

权重: 取决于请求中的LIMIT参数

LIMIT参数	权重
[1,100)	1

LIMIT参数	权重
[100, 500)	2
[500, 1000]	5
> 1000	10

参数:

名称	类型	是否必需	描述
pair	STRING	YES	标的交易对
contractType	ENUM	YES	合约类型
interval	ENUM	YES	时间间隔
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	默认值:500 最大值:1500

- 缺省返回最近的数据
- 合约类型:
 - PERPETUAL 永续合约
 - CURRENT_QUARTER 当季交割合约
 - NEXT_QUARTER 次季交割合约

价格指数K线数据

响应:

```
[
  [
    1591256400000,           // 开盘时间
    "9653.69440000",         // 开盘价
    "9653.69640000",         // 最高价
    "9651.38600000",         // 最低价
    "9651.55200000",         // 收盘价(当前K线未结束的即为最新价)
    "0",                      // 请忽略
    1591256459999,           // 收盘时间
    "0",                      // 请忽略
    60,                      // 请忽略
    "0",                      // 请忽略
    "0",                      // 请忽略
    "0"                       // 请忽略
  ]
]
```

GET /fapi/v1/indexPriceKlines

每根K线的开盘时间可视为唯一ID

权重: 取决于请求中的LIMIT参数

LIMIT参数	权重
[1,100)	1
[100, 500)	2
[500, 1000]	5
> 1000	10

参数:

名称	类型	是否必需	描述
pair	STRING	YES	标的交易对
interval	ENUM	YES	时间间隔
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	默认值:500 最大值:1500

- 缺省返回最近的数据

标记价格K线数据

响应:

```
[
  [
    [
      1591256400000,           // 开盘时间
      "9653.69440000",         // 开盘价
      "9653.69640000",         // 最高价
      "9651.38600000",         // 最低价
      "9651.55200000",         // 收盘价(当前K线未结束的即为最新价)
      "0",                      // 请忽略
      1591256459999,           // 收盘时间
      "0",                      // 请忽略
      60,                      // 请忽略
      "0",                      // 请忽略
      "0",                      // 请忽略
      "0"                       // 请忽略
    ]
  ]
]
```

GET /fapi/v1/markPriceKlines 每根K线的开盘时间可视为唯一ID

权重: 取决于请求中的LIMIT参数

LIMIT参数	权重

LIMIT参数	权重
[1,100)	1
[100, 500)	2
[500, 1000]	5
> 1000	10

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
interval	ENUM	YES	时间间隔
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	默认值:500 最大值:1500

- 缺省返回最近的数据

最新标记价格和资金费率

响应:

```
{
  "symbol": "BTCUSDT",          // 交易对
  "markPrice": "11793.63104562", // 标记价格
  "indexPrice": "11781.80495970", // 指数价格
  "estimatedSettlePrice": "11781.16138815", // 预估结算价,仅在交割开始前最后一小时有意义
  "lastFundingRate": "0.00038246", // 最近更新的资金费率
  "nextFundingTime": 1597392000000, // 下次资金费时间
  "interestRate": "0.00010000", // 标的资产基础利率
  "time": 1597370495002        // 更新时间
}
```

当不指定symbol时相应

```
[
  {
    "symbol": "BTCUSDT",          // 交易对
    "markPrice": "11793.63104562", // 标记价格
    "indexPrice": "11781.80495970", // 指数价格
    "estimatedSettlePrice": "11781.16138815", // 预估结算价,仅在交割开始前最后一小时有意义
    "lastFundingRate": "0.00038246", // 最近更新的资金费率
    "nextFundingTime": 1597392000000, // 下次资金费时间
    "interestRate": "0.00010000", // 标的资产基础利率
    "time": 1597370495002        // 更新时间
  }
]
```

```

    }
]
```

`GET /fapi/v1/premiumIndex`

采集各大交易所数据加权平均

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对

查询资金费率历史

响应:

```

[
  {
    "symbol": "BTCUSDT",           // 交易对
    "fundingRate": "-0.03750000", // 资金费率
    "fundingTime": 1570608000000, // 资金费时间
  },
  {
    "symbol": "BTCUSDT",
    "fundingRate": "0.00010000",
    "fundingTime": 1570636800000,
  }
]
```

`GET /fapi/v1/fundingRate`

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	默认值:100 最大值:1000

- 如果 `startTime` 和 `endTime` 都未发送, 返回最近 `limit` 条数据.
- 如果 `startTime` 和 `endTime` 之间的数据量大于 `limit`, 返回 `startTime` + `limit` 情况下的数据.

24hr价格变动情况

响应:

```
{
  "symbol": "BTCUSDT",
  "priceChange": "-94.9999800",      //24小时价格变动
  "priceChangePercent": "-95.960",    //24小时价格变动百分比
  "weightedAvgPrice": "0.29628482", //加权平均价
  "lastPrice": "4.00000200",        //最近一次成交价
  "lastQty": "200.00000000",       //最近一次成交额
  "openPrice": "99.00000000",      //24小时内第一次成交的价格
  "highPrice": "100.00000000",     //24小时最高价
  "lowPrice": "0.10000000",        //24小时最低价
  "volume": "8913.30000000",       //24小时成交量
  "quoteVolume": "15.30000000",    //24小时成交额
  "openTime": 1499783499040,       //24小时内，第一笔交易的发生时间
  "closeTime": 1499869899040,      //24小时内，最后一笔交易的发生时间
  "firstId": 28385,               //首笔成交id
  "lastId": 28460,                //末笔成交id
  "count": 76                    //成交笔数
}
```

或(当不发送交易对信息)

```
[
  {
    "symbol": "BTCUSDT",
    "priceChange": "-94.9999800",      //24小时价格变动
    "priceChangePercent": "-95.960",    //24小时价格变动百分比
    "weightedAvgPrice": "0.29628482", //加权平均价
    "lastPrice": "4.00000200",        //最近一次成交价
    "lastQty": "200.00000000",       //最近一次成交额
    "openPrice": "99.00000000",      //24小时内第一次成交的价格
    "highPrice": "100.00000000",     //24小时最高价
    "lowPrice": "0.10000000",        //24小时最低价
    "volume": "8913.30000000",       //24小时成交量
    "quoteVolume": "15.30000000",    //24小时成交额
    "openTime": 1499783499040,       //24小时内，第一笔交易的发生时间
    "closeTime": 1499869899040,      //24小时内，最后一笔交易的发生时间
    "firstId": 28385,               //首笔成交id
    "lastId": 28460,                //末笔成交id
    "count": 76                    //成交笔数
  }
]
```

`GET /fapi/v1/ticker/24hr`

请注意，不携带symbol参数会返回全部交易对数据，不仅数据庞大，而且权重极高

权重: * 带symbol为1 * 不带为40

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对

- 不发送交易对参数，则会返回所有交易对信息

最新价格

响应:

```
{
  "symbol": "LTCBTC",      // 交易对
  "price": "4.00000200",    // 价格
  "time": 1589437530011   // 撮合引擎时间
}
```

或(当不发送symbol)

```
[
  {
    "symbol": "BTCUSDT",    // 交易对
    "price": "6000.01",     // 价格
    "time": 1589437530011 // 撮合引擎时间
  }
]
```

`GET /fapi/v1/ticker/price`

返回最近价格

权重: * 单交易对① * 无交易对②

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对

- 不发送交易对参数，则会返回所有交易对信息

当前最优挂单

响应:

```
{
  "symbol": "BTCUSDT", // 交易对
  "bidPrice": "4.00000000", // 最优买单价
  "bidQty": "431.00000000", // 挂单量
  "askPrice": "4.00000200", // 最优卖单价
  "askQty": "9.00000000", // 挂单量
  "time": 1589437530011 // 撮合引擎时间
}
```

或(当不发送symbol)

```
[
  {
```

```

    "symbol": "BTCUSDT", // 交易对
    "bidPrice": "4.00000000", //最优买单价
    "bidQty": "431.00000000", //挂单量
    "askPrice": "4.00000200", //最优卖单价
    "askQty": "9.00000000", //挂单量
    "time": 1589437530011 // 撮合引擎时间
}
]

```

GET /fapi/v1/ticker/bookTicker

返回当前最优的挂单(最高买单，最低卖单)

权重: * 单交易对2

* 无交易对5

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对

- 不发送交易对参数，则会返回所有交易对信息

获取未平仓合约数

响应:

```
{
    "openInterest": "10659.509", // 未平仓合约数量
    "symbol": "BTCUSDT", // 交易对
    "time": 1589437530011 // 撮合引擎时间
}
```

GET /fapi/v1/openInterest

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对

合约持仓量

响应:

```
[
{
    "symbol": "BTCUSDT",
    "sumOpenInterest": "20403.12345678", // 持仓总数量
}
```

```

    "sumOpenInterestValue": "176196512.12345678", // 持仓总价值
    "timestamp": "1583127900000"

  },
  {

    "symbol": "BTCUSDT",
    "sumOpenInterest": "20401.36700000",
    "sumOpenInterestValue": "149940752.14464448",
    "timestamp": "1583128200000"

  },
]

```

GET /futures/data/openInterestHist

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	
period	ENUM	YES	"5m", "15m", "30m", "1h", "2h", "4h", "6h", "12h", "1d"
limit	LONG	NO	default 30, max 500
startTime	LONG	NO	
endTime	LONG	NO	

- 若无 starttime 和 endtime 限制，则默认返回当前时间往前的limit值
- 仅支持最近30天的数据

大户账户数多空比

响应:

```

[
  {
    "symbol": "BTCUSDT",
    "longShortRatio": "1.8105", // 大户多空账户数比值
    "longAccount": "0.6442", // 大户多仓账户数比例
    "shortAccount": "0.3558", // 大户空仓账户数比例
    "timestamp": "1583139600000"

  },
  {

    "symbol": "BTCUSDT",
    "longShortRatio": "1.8233",
    "longAccount": "0.5338",
    "shortAccount": "0.4662",
    "timestamp": "1583139600000"

  }
]

```

```

    "shortAccount": "0.3454",
    "timestamp": "1583139900000"
}
]

```

GET /futures/data/topLongShortAccountRatio

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	
period	ENUM	YES	"5m", "15m", "30m", "1h", "2h", "4h", "6h", "12h", "1d"
limit	LONG	NO	default 30, max 500
startTime	LONG	NO	
endTime	LONG	NO	

- 若无 starttime 和 endtime 限制，则默认返回当前时间往前的limit值
- 仅支持最近30天的数据

大户持仓量多空比

响应:

```

[
{
    "symbol": "BTCUSDT",
    "longShortRatio": "1.4342", // 大户多空持仓量比值
    "longAccount": "0.5344", // 大户多仓持仓量比例
    "shortAccount": "0.4238", // 大户空仓持仓量比例
    "timestamp": "1583139600000"
},
{
    "symbol": "BTCUSDT",
    "longShortRatio": "1.4337",
    "longAccount": "0.5891",
    "shortAccount": "0.4108",
    "timestamp": "1583139900000"
},
]

```

GET /futures/data/topLongShortPositionRatio

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	
period	ENUM	YES	"5m","15m","30m","1h","2h","4h","6h","12h","1d"
limit	LONG	NO	default 30, max 500
startTime	LONG	NO	
endTime	LONG	NO	

- 若无 starttime 和 endtime 限制，则默认返回当前时间往前的limit值
- 仅支持最近30天的数据

多空持仓人数比

响应:

```
[
  {
    "symbol": "BTCUSDT",
    "longShortRatio": "0.1960", // 多空人数比值
    "longAccount": "0.6622", // 多仓人数比例
    "shortAccount": "0.3378", // 空仓人数比例
    "timestamp": "1583139600000"
  },
  {
    "symbol": "BTCUSDT",
    "longShortRatio": "1.9559",
    "longAccount": "0.6617",
    "shortAccount": "0.3382",
    "timestamp": "1583139900000"
  }
]
```

GET /futures/data/globalLongShortAccountRatio

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	
period	ENUM	YES	"5m","15m","30m","1h","2h","4h","6h","12h","1d"

名称	类型	是否必需	描述
limit	LONG	NO	default 30, max 500
startTime	LONG	NO	
endTime	LONG	NO	

- 若无 starttime 和 endtime 限制，则默认返回当前时间往前的limit值
- 仅支持最近30天的数据

合约主动买卖量

响应:

```
[
  {
    "buySellRatio": "1.5586",
    "buyVol": "387.3300", // 主动买入量
    "sellVol": "248.5030", // 主动卖出量
    "timestamp": "1585614900000"

  },
  {
    "buySellRatio": "1.3104",
    "buyVol": "343.9290",
    "sellVol": "248.5030",
    "timestamp": "1583139900000"

  },
]
]
```

GET /futures/data/takerlongshortRatio

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	
period	ENUM	YES	"5m", "15m", "30m", "1h", "2h", "4h", "6h", "12h", "1d"
limit	LONG	NO	default 30, max 500
startTime	LONG	NO	
endTime	LONG	NO	

- 若无 starttime 和 endtime 限制，则默认返回当前时间往前的limit值
- 仅支持最近30天的数据

杠杆代币历史净值K线

响应:

```
[
  [
    1598371200000,      // 开盘时间
    "5.88275270",       // 开盘净值
    "6.03142087",       // 最高净值
    "5.85749741",       // 最低净值
    "5.99403551",       // 收盘净值(当前K线未结束的即为最新净值)
    "2.28602984",       // 收盘真实杠杆
    1598374799999,      // 收盘时间
    "0",                 // 请忽略
    6209,                // 净值更新笔数
    "14517.64507907",   // 请忽略
    "0",                 // 请忽略
    "0"                  // 请忽略
  ]
]
```

`GET /fapi/v1/lvtKlines`

杠杆代币净值系统基于合约架构，故该接口采用fapi

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	token name, e.g. "BTCDOWN", "BTCUP"
interval	ENUM	YES	
startTime	LONG	NO	
endTime	LONG	NO	
limit	INT	NO	默认 500, 最大 1000

- 如果未发送 startTime 和 endTime， 默认返回最近的交易

综合指数交易对信息

响应:

```
[
  {
    "symbol": "DEFIUSDT",
    "time": 1589437530011,      // 请求时间
    "component": "baseAsset",   // 成分资产
    "baseAssetList": [

```

```
{
    "baseAsset": "BAL", // 基础资产
    "quoteAsset": "USDT", // 报价资产
    "weightInQuantity": "1.04406228", // 权重(数量)
    "weightInPercentage": "0.02783900" // 权重(比例)
},
{
    "baseAsset": "BAND",
    "quoteAsset": "USDT",
    "weightInQuantity": "3.53782729",
    "weightInPercentage": "0.03935200"
}
]
}
```

获取交易对为综合指数的基础成分信息。

`GET /fapi/v1/indexInfo`

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	

多资产模式资产汇率指数

响应:

```
{
    "symbol": "ADAUSD",
    "time": 1635740268004,
    "index": "1.92957370",
    "bidBuffer": "0.10000000",
    "askBuffer": "0.10000000",
    "bidRate": "1.73661633",
    "askRate": "2.12253107",
    "autoExchangeBidBuffer": "0.05000000",
    "autoExchangeAskBuffer": "0.05000000",
    "autoExchangeBidRate": "1.83309501",
    "autoExchangeAskRate": "2.02605238"
}
```

或(当不发送交易对信息)

```
[
    {
        "symbol": "ADAUSD",
        "time": 1635740268004,
```

```

    "index": "1.92957370",
    "bidBuffer": "0.10000000",
    "askBuffer": "0.10000000",
    "bidRate": "1.73661633",
    "askRate": "2.12253107",
    "autoExchangeBidBuffer": "0.05000000",
    "autoExchangeAskBuffer": "0.05000000",
    "autoExchangeBidRate": "1.83309501",
    "autoExchangeAskRate": "2.02605238"
}
]

```

`GET /fapi/v1/assetIndex`

多资产模式资产汇率指数

权重: 带symbol为1, 不带为10

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	资产对

Websocket 行情推送

- 本篇所列出的所有wss接口，共有如下两种连接方式：

- 连接方式一：
 - Base Url : `wss://fstream.binance.com`
 - 订阅单一stream格式为 `/ws/<streamName>`
 - 组合streams的URL格式为 `/stream?streams=<streamName1>/<streamName2>/<streamName3>`
 - 连接样例：
`wss://fstream.binance.com/ws/bnbusdt@aggTrade`
`wss://fstream.binance.com/stream?streams=bnbusdt@aggTrade/btcusdt@markPrice`
- 连接方式二：
 - Base Url : `wss://fstream-auth.binance.com`
 - 订阅单一stream格式为 `/ws/<streamName>?listenKey=<validateListenKey>`
 - 组合streams的URL格式为 `/stream?streams=<streamName1>/<streamName2>/<streamName3>&listenKey=<validateListenKey>`
 - <validateListenKey>在建立连接时，必须为一个有效的listenKey**
 - 连接样例：
`wss://fstream-auth.binance.com/ws/btcusdt@markPrice?listenKey=XaEAKTsQSRLZAGH9tuIu37p1SRsdjm1AVBoNYPUIT1TAko1WI22PgmBMpI1rS8Yh`
`wss://fstream-auth.binance.com/stream?streams=btcusdt@markPrice@1s/bnbusdt@markPrice&listenKey=XaEAKTsQSRLZAGH9tuIu37p1SRsdjm1AVBoNYPUIT1TAko1WI22PgmBMpI1rS8Yh`

- 订阅组合streams时，事件payload会以这样的格式封装 `{"stream":"<streamName>","data":<rawPayload>}`
- stream名称中所有交易对均为**小写**。
- 每个链接有效期不超过24小时，请妥善处理断线重连。
- 服务端每5分钟会发送ping帧，客户端应当在15分钟内回复pong帧，否则服务端会主动断开链接。允许客户端发送不成对的pong帧(即客户端可以以高于15分钟每次的频率发送pong帧保持链接)。
- Websocket服务器每秒最多接受10个订阅消息。

- 如果用户发送的消息超过限制，连接会被断开连接。反复被断开连接的IP有可能被服务器屏蔽。
- 单个连接最多可以订阅 **200** 个Streams。

实时订阅/取消数据流

- 以下数据可以通过websocket发送以实现订阅或取消订阅数据流。示例如下。
- 响应内容中的 id 是无符号整数，作为往来信息的唯一标识。

订阅一个信息流

响应

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

- 请求

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

取消订阅一个信息流

响应

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

- 请求

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

已订阅信息流

响应

```
{
  "result": [
```

```

    "btcusdt@aggTrade"
],
"id": 3
}

```

- **请求**

```
{
"method": "LIST_SUBSCRIPTIONS",
"id": 3
}
```

设定属性

当前，唯一可以设置的属性是设置是否启用combined("组合")信息流。

当使用 /ws/ ("原始信息流") 进行连接时，combined 属性设置为 false，而使用 /stream/ 进行连接时则将属性设置为 true。

响应

```

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

```

- **请求**

```
{
"method": "SET_PROPERTY",
"params": [
  "combined",
  true
],
"id": 5
}
```

检索属性

响应

```

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

```

- **请求**

```
{
"method": "GET_PROPERTY",
"params": [
  "combined"
],
"id": 2
}
```

错误信息

错误信息	描述
{"code": 0, "msg": "Unknown property"}	<code>SET_PROPERTY</code> 或 <code>GET_PROPERTY</code> 中应用的参数无效
{"code": 1, "msg": "Invalid value type: expected Boolean"}	仅接受 <code>true</code> 或 <code>false</code>
{"code": 2, "msg": "Invalid request: property name must be a string"}	提供的属性名无效
{"code": 2, "msg": "Invalid request: request ID must be an unsigned integer"}	参数 <code>id</code> 未提供或 <code>id</code> 值是无效类型
{"code": 2, "msg": "Invalid request: unknown variant %s, expected one of <code>SUBSCRIBE</code> , <code>UNSUBSCRIBE</code> , <code>LIST_SUBSCRIPTIONS</code> , <code>SET_PROPERTY</code> , <code>GET_PROPERTY</code> at line 1 column 28"}	错字提醒，或提供的值不是预期类型
{"code": 2, "msg": "Invalid request: too many parameters"}	数据中提供了不必要参数
{"code": 2, "msg": "Invalid request: property name must be a string"}	未提供属性名
{"code": 2, "msg": "Invalid request: missing field <code>method</code> at line 1 column 73"}	数据未提供 <code>method</code>
{"code": 3, "msg": "Invalid JSON: expected value at line %s column %s"}	JSON 语法有误。

最新合约价格

aggTrade 中的价格'p'或ticker/miniTicker中的价格'c'均可以作为最新成交价。

归集交易

Payload:

```
{
  "e": "aggTrade", // 事件类型
  "E": 123456789, // 事件时间
  "s": "BNBUSDT", // 交易对
  "a": 5933014, // 归集成交 ID
  "p": "0.001", // 成交价格
  "q": "100", // 成交量
  "f": 100, // 被归集的首个交易 ID
  "l": 105, // 被归集的末次交易 ID
  "T": 123456785, // 成交时间
  "m": true // 买方是否是做市方。如 true，则此次成交是一个主动卖出单，否则是一个主动买单。
}
```

同一价格、同一方向、同一时间(100ms计算)的trade会被聚合为一条。

Stream Name:

<symbol>@aggTrade

Update Speed: 100ms

最新标记价格

Payload:

```
{
  "e": "markPriceUpdate",      // 事件类型
  "E": 1562305380000,         // 事件时间
  "s": "BTCUSDT",             // 交易对
  "p": "11794.15000000",       // 标记价格
  "i": "11784.62659091",       // 现货指指数价格
  "P": "11784.25641265",       // 预估结算价,仅在结算前最后一小时有参考价值
  "r": "0.00038167",           // 资金费率
  "T": 1562306400000          // 下次资金时间
}
```

Stream Name:

<symbol>@markPrice 或 <symbol>@markPrice@1s

Update Speed: 3000ms 或 1000ms

全市场最新标记价格

Payload:

```
[
  [
    {
      "e": "markPriceUpdate",      // 事件类型
      "E": 1562305380000,         // 事件时间
      "s": "BTCUSDT",             // 交易对
      "p": "11185.87786614",       // 标记价格
      "i": "11784.62659091",       // 现货指指数价格
      "P": "11784.25641265",       // 预估结算价,仅在结算前最后一小时有参考价值
      "r": "0.00030000",           // 资金费率
      "T": 1562306400000          // 下个资金时间
    }
  ]
]
```

Stream Name:

!markPrice@arr 或 !markPrice@arr@1s

Update Speed: 3000ms 或 1000ms

K线

Payload:

```
{
  "e": "kline",      // 事件类型
```

```

"E": 123456789,    // 事件时间
"s": "BNBUSDT",    // 交易对
"k": {
  "t": 123400000, // 这根K线的起始时间
  "T": 123460000, // 这根K线的结束时间
  "s": "BNBUSDT", // 交易对
  "i": "1m",      // K线间隔
  "f": 100,        // 这根K线期间第一笔成交ID
  "L": 200,        // 这根K线期间末一笔成交ID
  "o": "0.0010",  // 这根K线期间第一笔成交价
  "c": "0.0020",  // 这根K线期间末一笔成交价
  "h": "0.0025",  // 这根K线期间最高成交价
  "l": "0.0015",  // 这根K线期间最低成交价
  "v": "1000",    // 这根K线期间成交量
  "n": 100,        // 这根K线期间成交笔数
  "x": false,      // 这根K线是否完结(是否已经开始下一根K线)
  "q": "1.0000",  // 这根K线期间成交额
  "V": "500",     // 主动买入的成交量
  "Q": "0.500",   // 主动买入的成交额
  "B": "123456"   // 忽略此参数
}
}

```

K线stream逐秒推送所请求的K线种类(最新一根K线)的更新。推送间隔250毫秒(如有刷新)

订阅Kline需要提供间隔参数，最短为分钟线，最长为月线。支持以下间隔：

m -> 分钟; h -> 小时; d -> 天; w -> 周; M -> 月

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

Stream Name:

<symbol>@kline_<interval>

Update Speed: 250ms

连续合约K线

Payload:

```
{
  "e": "continuous_kline", // 事件类型
  "E": 1607443058651,    // 事件时间
}
```

```

"ps": "BTCUSDT",           // 标的交易对
"ct": "PERPETUAL",         // 合约类型
"k": {
  "t": 1607443020000,      // 这根K线的起始时间
  "T": 1607443079999,      // 这根K线的结束时间
  "i": "1m",                // K线间隔
  "f": 116467658886,       // 这根K线期间第一笔成交ID
  "L": 116468012423,       // 这根K线期间末一笔成交ID
  "o": "18787.00",          // 这根K线期间第一笔成交价
  "c": "18804.04",          // 这根K线期间末一笔成交价
  "h": "18804.04",          // 这根K线期间最高成交价
  "l": "18786.54",          // 这根K线期间最低成交价
  "v": "197.664",           // 这根K线期间成交量
  "n": 543,                 // 这根K线期间成交笔数
  "x": false,                // 这根K线是否完结(是否已经开始下一根K线)
  "q": "3715253.19494",     // 这根K线期间成交额
  "V": "184.769",            // 主动买入的成交量
  "Q": "3472925.84746",     // 主动买入的成交额
  "B": "0"                   // 忽略此参数
}
}

```

K线stream逐秒推送所请求的K线种类(最新一根K线)的更新。

合约类型:

- perpetual 永续合约
- current_quarter 当季交割合约
- next_quarter 次季交割合约

订阅Kline需要提供间隔参数,最短为分钟线,最长为月线。支持以下间隔:

m -> 分钟; h -> 小时; d -> 天; w -> 周; M -> 月

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

Stream Name:

<pair>_<contractType>@continuousKline_<interval>

Update Speed: 250ms

按Symbol的精简Ticker

Payload:

```
{
  "e": "24hrMiniTicker", // 事件类型
  "E": 123456789, // 事件时间(毫秒)
  "s": "BNBUSDT", // 交易对
  "c": "0.0025", // 最新成交价格
  "o": "0.0010", // 24小时前开始第一笔成交价格
  "h": "0.0025", // 24小时内最高成交价
  "l": "0.0010", // 24小时内最低成交价
  "v": "10000", // 成交量
  "q": "18" // 成交额
}
```

按Symbol刷新的24小时精简ticker信息.

Stream Name:

<symbol>@miniTicker

Update Speed: 500ms

全市场的精简Ticker

Payload:

```
[
  [
    {
      "e": "24hrMiniTicker", // 事件类型
      "E": 123456789, // 事件时间(毫秒)
      "s": "BNBUSDT", // 交易对
      "c": "0.0025", // 最新成交价格
      "o": "0.0010", // 24小时前开始第一笔成交价格
      "h": "0.0025", // 24小时内最高成交价
      "l": "0.0010", // 24小时内最低成交价
      "v": "10000", // 成交量
      "q": "18" // 成交额
    }
  ]
]
```

所有symbol24小时精简ticker信息.需要注意的是，只有发生变化的ticker更新才会被推送。

Stream Name:

!miniTicker@arr

Update Speed: 1000ms

按Symbol的完整Ticker

Payload:

```
{
  "e": "24hrTicker", // 事件类型
```

```

"E": 123456789,      // 事件时间
"s": "BNBUSDT",       // 交易对
"p": "0.0015",        // 24小时价格变化
"P": "250.00",        // 24小时价格变化(百分比)
"w": "0.0018",        // 平均价格
"c": "0.0025",        // 最新成交价格
"Q": "10",             // 最新成交价格上的成交量
"o": "0.0010",        // 24小时内第一笔成交的价格
"h": "0.0025",        // 24小时内最高成交价
"l": "0.0010",        // 24小时内最低成交价
"v": "10000",          // 24小时内成交量
"q": "18",              // 24小时内成交额
"O": 0,                // 统计开始时间
"C": 86400000,         // 统计关闭时间
"F": 0,                // 24小时内第一笔成交交易ID
"L": 18150,             // 24小时内最后一笔成交交易ID
"n": 18151             // 24小时内成交数
}

```

按Symbol刷新的24小时完整ticker信息

Stream Name:

<symbol>@ticker

Update Speed: 500ms

全市场的完整Ticker

Payload:

```

[
{
  "e": "24hrTicker",   // 事件类型
  "E": 123456789,     // 事件时间
  "s": "BNBUSDT",      // 交易对
  "p": "0.0015",        // 24小时价格变化
  "P": "250.00",        // 24小时价格变化(百分比)
  "w": "0.0018",        // 平均价格
  "c": "0.0025",        // 最新成交价格
  "Q": "10",             // 最新成交价格上的成交量
  "o": "0.0010",        // 24小时内第一笔成交的价格
  "h": "0.0025",        // 24小时内最高成交价
  "l": "0.0010",        // 24小时内最低成交价
  "v": "10000",          // 24小时内成交量
  "q": "18",              // 24小时内成交额
  "O": 0,                // 统计开始时间
  "C": 86400000,         // 统计结束时间
  "F": 0,                // 24小时内第一笔成交交易ID
  "L": 18150,             // 24小时内最后一笔成交交易ID
  "n": 18151             // 24小时内成交数
}
]
```

所有symbol 24小时完整ticker信息.需要注意的是，只有发生变化的ticker更新才会被推送。

Stream Name:

!ticker@arr

Update Speed: 1000ms

按Symbol的最优挂单信息

Payload:

```
{  
    "e": "bookTicker",           // 事件类型  
    "u": 400900217,             // 更新ID  
    "E": 1568014460893,         // 事件推送时间  
    "T": 1568014460891,         // 撮合时间  
    "s": "BNBUSDT",              // 交易对  
    "b": "25.35190000",          // 买单最优挂单价格  
    "B": "31.21000000",          // 买单最优挂单数量  
    "a": "25.36520000",          // 卖单最优挂单价格  
    "A": "40.66000000"           // 卖单最优挂单数量  
}
```

实时推送指定交易对最优挂单信息

Stream Name: <symbol>@bookTicker

Update Speed: 实时

全市场最优挂单信息

Payload:

```
{  
    // Same as <symbol>@bookTicker payload  
}  
}
```

所有交易对交易对最优挂单信息

Stream Name: !bookTicker

Update Speed: 实时

强平订单

Payload:

```
{  
  "e": "forceOrder",  
  // 事件类型
```

```

"E":1568014460893,           // 事件时间
{o}:

  "s":"BTCUSDT",             // 交易对
  "S":"SELL",                // 订单方向
  "o":"LIMIT",               // 订单类型
  "f":"IOC",                 // 有效方式
  "q":"0.014",                // 订单数量
  "p":"9910",                // 订单价格
  "ap":"9910",               // 平均价格
  "X":"FILLED",              // 订单状态
  "l":"0.014",                // 订单最近成交量
  "z":"0.014",                // 订单累计成交量
  "T":1568014460893,         // 交易时间

}

}

```

推送特定symbol的强平订单快照信息。

1000ms内至多仅推送一条最近的强平订单作为快照

Stream Name: <symbol>@forceOrder

Update Speed: 1000ms

全市场强平订单

Payload:

```

{

  "e":"forceOrder",          // 事件类型
  "E":1568014460893,        // 事件时间
  "o":{

    "s":"BTCUSDT",           // 交易对
    "S":"SELL",                // 订单方向
    "o":"LIMIT",               // 订单类型
    "f":"IOC",                 // 有效方式
    "q":"0.014",                // 订单数量
    "p":"9910",                // 订单价格
    "ap":"9910",               // 平均价格
    "X":"FILLED",              // 订单状态
    "l":"0.014",                // 订单最近成交量
    "z":"0.014",                // 订单累计成交量
    "T":1568014460893,         // 交易时间

  }

}

```

推送全市场强平订单快照信息

每个symbol，1000ms内至多仅推送一条最近的强平订单作为快照

Stream Name: !forceOrder@arr

Update Speed: 实时

有限档深度信息

Payload:

```
{
  "e": "depthUpdate",           // 事件类型
  "E": 1571889248277,          // 事件时间
  "T": 1571889248276,          // 交易时间
  "s": "BTCUSDT",              // 合约
  "U": 390497796,              // 买方
  "u": 390497878,
  "pu": 390497794,
  "b": [                         // 买方
    [
      "7403.89",                // 价格
      "0.002"                   // 数量
    ],
    [
      "7403.90",
      "3.906"
    ],
    [
      "7404.00",
      "1.428"
    ],
    [
      "7404.85",
      "5.239"
    ],
    [
      "7405.43",
      "2.562"
    ]
  ],
  "a": [                         // 卖方
    [
      "7405.96",                // 价格
      "3.340"                   // 数量
    ],
    [
      "7406.63",
      "4.525"
    ],
    [
      "7407.08",
      "2.475"
    ],
    [
      "7407.15"
    ]
  ]
}
```

```

    "4.800"
],
[
  "7407.20",
  "0.175"
]
}

```

推送有限档深度信息。levels表示几档买卖单信息, 可选 5/10/20档

Stream Names: <symbol>@depth<levels> 或 <symbol>@depth<levels>@500ms 或 <symbol>@depth<levels>@100ms.

Update Speed: 250ms 或 500ms 或 100ms

增量深度信息

Payload:

```

{
  "e": "depthUpdate",      // 事件类型
  "E": 123456789,         // 事件时间
  "T": 123456788,         // 摄合时间
  "s": "BNBUSDT",          // 交易对
  "U": 157,                // 从上次推送至今新增的第一个 update Id
  "u": 160,                // 从上次推送至今新增的最后一个 update Id
  "pu": 149,                // 上次推送的最后一个update Id(即上条消息的‘u’)
  "b": [
    [
      "0.0024",             // 价格
      "10"                  // 数量
    ]
  ],
  "a": [
    [
      "0.0026",             // 价格
      "100"                 // 数量
    ]
  ]
}

```

orderbook的变化部分，推送间隔250毫秒,500毫秒，100毫秒(如有刷新)

Stream 名称:

<symbol>@depth OR <symbol>@depth@500ms OR <symbol>@depth@100ms

Update Speed: 250ms 或 500ms 或 100ms

如何正确在本地维护一个orderbook副本

1. 订阅 `wss://fstream.binance.com/stream?streams=btcusdt@depth`
2. 开始缓存收到的更新。同一个价位，后收到的更新覆盖前面的。

3. 访问Rest接口 <https://fapi.binance.com/fapi/v1/depth?symbol=BTCUSDT&limit=1000>获得一个1000档的深度快照
4. 将目前缓存到的信息中 $u < \text{步骤3中获取到的快照中的} \text{lastUpdateId}$ 的部分丢弃(丢弃更早的信息，已经过期)。
5. 将深度快照中的内容更新到本地orderbook副本中，并从websocket接收到的第一个 $u \leq \text{lastUpdateId}$ 且 $u \geq \text{lastUpdateId}$ 的event开始继续更新本地副本。
6. 每一个新event的 p_u 应该等于上一个event的 u ，否则可能出现了丢包，请从step3重新进行初始化。
7. 每一个event中的挂单量代表这个价格目前的挂单量**绝对值**，而不是相对变化。
8. 如果某个价格对应的挂单量为0，表示该价位的挂单已经撤单或者被吃，应该移除这个价位。

综合指数交易对信息流

Payload:

```
{
  "e": "compositeIndex",           // 事件类型
  "E": 1602310596000,             // 事件事件
  "s": "DEFIUSDT",                // 交易对
  "p": "554.41604065",            // 价格
  "C": "baseAsset",
  "c": [                           // 成分信息
    {
      "b": "BAL",                  // 基础资产
      "q": "USDT",                 // 报价资产
      "w": "1.04884844",            // 权重(数量)
      "W": "0.01457800",            // 权重(比例)
      "i": "24.33521021"           // 指数价格
    },
    {
      "b": "BAND",
      "q": "USDT",
      "w": "3.53782729",
      "W": "0.03935200",
      "i": "7.26420084"
    }
  ]
}
```

获取交易对为综合指数的基础成分信息。 推送间隔1000毫秒(如有刷新)

Stream Name: <symbol>@compositeIndex

Update Speed: 1000ms

账户和交易接口

- 考虑到剧烈行情下, RESTful接口可能存在查询延迟，我们强烈建议您优先从Websocket user data stream推送的消息来获取订单，成交，仓位等信息。

划转

执行现货账户与合约账户之间的划转, 详情请见[这里](#).

获取划转历史

获取现货账户与合约账户之间的资金划转历史记录, 详情请见[这里](#).

更改持仓模式(TRADE)

响应:

```
{
  "code": 200,
  "msg": "success"
}
```

`POST /fapi/v1/positionSide/dual (HMAC SHA256)`

变换用户在 **所有symbol** 合约上的持仓模式 : 双向持仓或单向持仓。

权重: 1

参数:

名称	类型	是否必需	描述
dualSidePosition	STRING	YES	"true": 双向持仓模式 ; "false": 单向持仓模式
recvWindow	LONG	NO	
timestamp	LONG	YES	

查询持仓模式(USER_DATA)

响应:

```
{
  "dualSidePosition": true // "true": 双向持仓模式; "false": 单向持仓模式
}
```

`GET /fapi/v1/positionSide/dual (HMAC SHA256)`

查询用户目前在 **所有symbol** 合约上的持仓模式 : 双向持仓或单向持仓。

权重: 30

参数:

名称	类型	是否必需	描述
recvWindow	LONG	NO	
timestamp	LONG	YES	

更改联合保证金模式(TRADE)

响应:

```
{
  "code": 200,
  "msg": "success"
}
```

POST /fapi/v1/multiAssetsMargin (HMAC SHA256)

变换用户在 **所有symbol** 合约上的联合保证金模式：开启或关闭联合保证金模式。

权重: 1

参数:

名称	类型	是否必需	描述
multiAssetsMargin	STRING	YES	"true": 联合保证金模式开启；"false": 联合保证金模式关闭
recvWindow	LONG	NO	
timestamp	LONG	YES	

查询联合保证金模式(USER_DATA)

响应:

```
{
  "multiAssetsMargin": true // "true": 联合保证金模式开启；"false": 联合保证金模式关闭
}
```

GET /fapi/v1/multiAssetsMargin (HMAC SHA256)

查询用户目前在 **所有symbol** 合约上的联合保证金模式。

权重: 30

参数:

名称	类型	是否必需	描述

名称	类型	是否必需	描述
recvWindow	LONG	NO	
timestamp	LONG	YES	

下单 (TRADE)

响应:

```
{
  "clientOrderId": "testOrder", // 用户自定义的订单号
  "cumQty": "0",
  "cumQuote": "0", // 成交金额
  "executedQty": "0", // 成交量
  "orderId": 22542179, // 系统订单号
  "avgPrice": "0.00000", // 平均成交价
  "origQty": "10", // 原始委托数量
  "price": "0", // 委托价格
  "reduceOnly": false, // 仅减仓
  "side": "SELL", // 买卖方向
  "positionSide": "SHORT", // 持仓方向
  "status": "NEW", // 订单状态
  "stopPrice": "0", // 触发价, 对`TRAILING_STOP_MARKET`无效
  "closePosition": false, // 是否条件全平仓
  "symbol": "BTCUSDT", // 交易对
  "timeInForce": "GTC", // 有效方法
  "type": "TRAILING_STOP_MARKET", // 订单类型
  "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
  "activatePrice": "9020", // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET`订单返回此字段
  "priceRate": "0.3", // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET`订单返回此字段
  "updateTime": 1566818724722, // 更新时间
  "workingType": "CONTRACT_PRICE", // 条件价格触发类型
  "priceProtect": false // 是否开启条件单触发保护
}
```

POST /fapi/v1/order (HMAC SHA256)

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
side	ENUM	YES	买卖方向 <code>SELL</code> , <code>BUY</code>
positionSide	ENUM	NO	持仓方向, 单向持仓模式下非必填, 默认且仅可填 <code>BOTH</code> ; 在双向持仓模式下必填, 且仅可选择 <code>LONG</code> 或 <code>SHORT</code>
type	ENUM	YES	订单类型 <code>LIMIT</code> , <code>MARKET</code> , <code>STOP</code> , <code>TAKE_PROFIT</code> , <code>STOP_MARKET</code> , <code>TAKE_PROFIT_MARKET</code> , <code>TRAILING_STOP_MARKET</code>

名称	类型	是否必需	描述
reduceOnly	STRING	NO	<code>true</code> , <code>false</code> ; 非双开模式下默认 <code>false</code> ; 双开模式下不接受此参数; 使用 <code>closePosition</code> 不支持此参数。
quantity	DECIMAL	NO	下单数量, 使用 <code>closePosition</code> 不支持此参数。
price	DECIMAL	NO	委托价格
newClientOrderId	STRING	NO	用户自定义的订单号, 不可以重复出现在挂单中。如空缺系统会自动赋值。必须满足正则规则 <code>^[\.\w\-_]{1,36}\$</code>
stopPrice	DECIMAL	NO	触发价, 仅 <code>STOP</code> , <code>STOP_MARKET</code> , <code>TAKE_PROFIT</code> , <code>TAKE_PROFIT_MARKET</code> 需要此参数
closePosition	STRING	NO	<code>true</code> , <code>false</code> ; 触发后全部平仓, 仅支持 <code>STOP_MARKET</code> 和 <code>TAKE_PROFIT_MARKET</code> ; 不与 <code>quantity</code> 合用; 自带只平仓效果, 不与 <code>reduceOnly</code> 合用
activationPrice	DECIMAL	NO	追踪止损激活价格, 仅 <code>TRAILING_STOP_MARKET</code> 需要此参数, 默认为下单当前市场价格(支持不同 <code>workingType</code>)
callbackRate	DECIMAL	NO	追踪止损回调比例, 可取值范围[0.1, 5], 其中1代表1%, 仅 <code>TRAILING_STOP_MARKET</code> 需要此参数
timeInForce	ENUM	NO	有效方法
workingType	ENUM	NO	<code>stopPrice</code> 触发类型: <code>MARK_PRICE</code> (标记价格), <code>CONTRACT_PRICE</code> (合约最新价). 默认 <code>CONTRACT_PRICE</code>
priceProtect	STRING	NO	条件单触发保护: "TRUE", "FALSE", 默认"FALSE". 仅 <code>STOP</code> , <code>STOP_MARKET</code> , <code>TAKE_PROFIT</code> , <code>TAKE_PROFIT_MARKET</code> 需要此参数
newOrderRespType	ENUM	NO	"ACK", "RESULT", 默认"ACK"
recvWindow	LONG	NO	
timestamp	LONG	YES	

根据 order `type` 的不同, 某些参数强制要求, 具体如下:

Type	强制要求的参数
<code>LIMIT</code>	<code>timeInForce</code> , <code>quantity</code> , <code>price</code>
<code>MARKET</code>	<code>quantity</code>
<code>STOP</code> , <code>TAKE_PROFIT</code>	<code>quantity</code> , <code>price</code> , <code>stopPrice</code>
<code>STOP_MARKET</code> , <code>TAKE_PROFIT_MARKET</code>	<code>stopPrice</code>
<code>TRAILING_STOP_MARKET</code>	<code>callbackRate</code>

- 条件单的触发必须:

- 如果订单参数`priceProtect`为`true`:
 - 达到触发价时, `MARK_PRICE`(标记价格)与`CONTRACT_PRICE`(合约最新价)之间的价差不能超过改symbol触发保护阈值

- 触发保护阈值请参考接口 `GET /fapi/v1/exchangeInfo` 返回内容相应symbol中"triggerProtect"字段
- `STOP`, `STOP_MARKET` 止损单:
 - 买入: 最新合约价格/标记价格高于等于触发价 `stopPrice`
 - 卖出: 最新合约价格/标记价格低于等于触发价 `stopPrice`
- `TAKE_PROFIT`, `TAKE_PROFIT_MARKET` 止盈单:
 - 买入: 最新合约价格/标记价格低于等于触发价 `stopPrice`
 - 卖出: 最新合约价格/标记价格高于等于触发价 `stopPrice`
- `TRAILING_STOP_MARKET` 跟踪止损单:
 - 买入: 当合约价格/标记价格区间最低价格低于激活价格 `activationPrice` 且最新合约价格/标记价高于等于最低价设定回调幅度。
 - 卖出: 当合约价格/标记价格区间最高价格高于激活价格 `activationPrice` 且最新合约价格/标记价低于等于最高价设定回调幅度。
- `TRAILING_STOP_MARKET` 跟踪止损单如果遇到报错 `{"code": -2021, "msg": "Order would immediately trigger."}` 表示订单不满足以下条件:
 - 买入: 指定的 `activationPrice` 必须小于 latest price
 - 卖出: 指定的 `activationPrice` 必须大于 latest price
- `newOrderRespType` 如果传 `RESULT`:
 - `MARKET` 订单将直接返回成交结果 ;
 - 配合使用特殊 `timeInForce` 的 `LIMIT` 订单将直接返回成交或过期拒绝结果。
- `STOP_MARKET`, `TAKE_PROFIT_MARKET` 配合 `closePosition=true`:
 - 条件单触发依照上述条件单触发逻辑
 - 条件触发后, 平掉当时持有所有多头仓位(若为买单)或当时持有所有空头仓位(若为买单)
 - 不支持 `quantity` 参数
 - 自带只平仓属性, 不支持 `reduceOnly` 参数
 - 双开模式下, `LONG` 方向上不支持 `BUY`; `SHORT` 方向上不支持 `SELL`

测试下单接口 (TRADE)

响应:

字段与下单接口一致, 但均为无效值

`POST /fapi/v1/order/test (HMAC SHA256)`

用于测试订单请求, 但不会提交到撮合引擎

参数:

参考 `POST /fapi/v1/order`

批量下单 (TRADE)

响应:

```
[  
{  
  "clientOrderId": "testOrder", // 用户自定义的订单号
```

```

    "cumQty": "0",
    "cumQuote": "0", // 成交金额
    "executedQty": "0", // 成交量
    "orderId": 22542179, // 系统订单号
    "avgPrice": "0.00000", // 平均成交价
    "origQty": "10", // 原始委托数量
    "price": "0", // 委托价格
    "reduceOnly": false, // 仅减仓
    "side": "SELL", // 买卖方向
    "positionSide": "SHORT", // 持仓方向
    "status": "NEW", // 订单状态
    "stopPrice": "0", // 触发价, 对`TRAILING_STOP_MARKET`无效
    "closePosition": false, // 是否条件全平仓
    "symbol": "BTCUSDT", // 交易对
    "timeInForce": "GTC", // 有效方法
    "type": "TRAILING_STOP_MARKET", // 订单类型
    "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
    "activatePrice": "9020", // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET`订单返回此字段
    "priceRate": "0.3", // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET`订单返回此字段
    "updateTime": 1566818724722, // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false // 是否开启条件单触发保护
},
{
    "code": -2022,
    "msg": "ReduceOnly Order is rejected."
}
]

```

POST /fapi/v1/batchOrders (HMAC SHA256)

权重: 5

参数:

名称	类型	是否必需	描述
batchOrders	list	YES	订单列表, 最多支持5个订单
recvWindow	LONG	NO	
timestamp	LONG	YES	

其中 **batchOrders** 应以 **list of JSON** 格式填写订单参数

- 例子: /fapi/v1/batchOrders?batchOrders=[{"type":"LIMIT","timeInForce":"GTC","symbol":"BTCUSDT","side":"BUY","price":"10001","quantity":"0.001"}]

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
side	ENUM	YES	买卖方向 SELL , BUY
positionSide	ENUM	NO	持仓方向, 单向持仓模式下非必填, 默认且仅可填 BOTH ; 在双向持仓模式下必填, 且仅可选择 LONG 或 SHORT

名称	类型	是否必需	描述
type	ENUM	YES	订单类型 <code>LIMIT</code> , <code>MARKET</code> , <code>STOP</code> , <code>TAKE_PROFIT</code> , <code>STOP_MARKET</code> , <code>TAKE_PROFIT_MARKET</code> , <code>TRAILING_STOP_MARKET</code>
reduceOnly	STRING	NO	<code>true</code> , <code>false</code> ; 非双开模式下默认 <code>false</code> ; 双开模式下不接受此参数。
quantity	DECIMAL	YES	下单数量
price	DECIMAL	NO	委托价格
newClientOrderId	STRING	NO	用户自定义的订单号，不可以重复出现在挂单中。如空缺系统会自动赋值. 必须满足正则规则 <code>^[\.\w\-_]{1,36}\\$</code>
stopPrice	DECIMAL	NO	触发价, 仅 <code>STOP</code> , <code>STOP_MARKET</code> , <code>TAKE_PROFIT</code> , <code>TAKE_PROFIT_MARKET</code> 需要此参数
activationPrice	DECIMAL	NO	追踪止损激活价格 , 仅 <code>TRAILING_STOP_MARKET</code> 需要此参数, 默认为下单当前市场价格 (支持不同 <code>workingType</code>)
callbackRate	DECIMAL	NO	追踪止损回调比例 , 可取值范围[0.1, 4], 其中 1代表1% ,仅 <code>TRAILING_STOP_MARKET</code> 需要此参数
timeInForce	ENUM	NO	有效方法
workingType	ENUM	NO	stopPrice 触发类型: <code>MARK_PRICE</code> (标记价格), <code>CONTRACT_PRICE</code> (合约最新价). 默认 <code>CONTRACT_PRICE</code>
priceProtect	STRING	NO	条件单触发保护 : "TRUE", "FALSE", 默认 "FALSE". 仅 <code>STOP</code> , <code>STOP_MARKET</code> , <code>TAKE_PROFIT</code> , <code>TAKE_PROFIT_MARKET</code> 需要此参数
newOrderRespType	ENUM	NO	"ACK", "RESULT", 默认 "ACK"

- 具体订单条件规则 , 与普通下单一致
- 批量下单采取并发处理 , 不保证订单撮合顺序
- 批量下单的返回内容顺序 , 与订单列表顺序一致

查询订单 (USER_DATA)

响应:

```
{
  "avgPrice": "0.00000",           // 平均成交价
  "clientOrderId": "abc",          // 用户自定义的订单号
  "cumQuote": "0",                // 成交金额
  "executedQty": "0",              // 成交量
  "orderId": 1573346959,           // 系统订单号
  "origQty": "0.40",              // 原始委托数量
  "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
  "price": "0",                   // 委托价格
  "reduceOnly": false,             // 是否仅减仓
  "side": "BUY",                 // 买卖方向
  "positionSide": "SHORT",         // 持仓方向
  "status": "NEW",                // 订单状态
  "stopPrice": "9300",             // 触发价, 对`TRAILING_STOP_MARKET` 无效
}
```

```

    "closePosition": false, // 是否条件全平仓
    "symbol": "BTCUSDT", // 交易对
    "time": 1579276756075, // 订单时间
    "timeInForce": "GTC", // 有效方法
    "type": "TRAILING_STOP_MARKET", // 订单类型
    "activatePrice": "9020", // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "priceRate": "0.3", // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "updateTime": 1579276756075, // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false // 是否开启条件单触发保护
}

```

GET /fapi/v1/order (HMAC SHA256)

查询订单状态

- 请注意，如果订单满足如下条件，不会被查询到：
 - 订单的最终状态为 `CANCELED` 或者 `EXPIRED`, 并且
 - 订单没有任何的成交记录, 并且
 - 订单生成时间 + 7天 < 当前时间

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
orderId	LONG	NO	系统订单号
origClientOrderId	STRING	NO	用户自定义的订单号
recvWindow	LONG	NO	
timestamp	LONG	YES	

注意:

- 至少需要发送 `orderId` 与 `origClientOrderId` 中的一个

撤销订单 (TRADE)

响应:

```

{
    "clientOrderId": "myOrder1", // 用户自定义的订单号
    "cumQty": "0",
    "cumQuote": "0", // 成交金额
    "executedQty": "0", // 成交量
    "orderId": 283194212, // 系统订单号
    "origQty": "11", // 原始委托数量
    "price": "0", // 委托价格
    "reduceOnly": false, // 仅减仓
    "side": "BUY", // 买卖方向
    "positionSide": "SHORT", // 持仓方向
}

```

```

    "status": "CANCELED", // 订单状态
    "stopPrice": "9300", // 触发价, 对`TRAILING_STOP_MARKET`无效
    "closePosition": false, // 是否条件全平仓
    "symbol": "BTCUSDT", // 交易对
    "timeInForce": "GTC", // 有效方法
    "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
    "type": "TRAILING_STOP_MARKET", // 订单类型
    "activatePrice": "9020", // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET`订单返回此字段
    "priceRate": "0.3", // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET`订单返回此字段
    "updateTime": 1571110484038, // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false // 是否开启条件单触发保护
}

```

`DELETE /fapi/v1/order (HMAC SHA256)`

权重: 1

Parameters:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
orderId	LONG	NO	系统订单号
origClientOrderId	STRING	NO	用户自定义的订单号
recvWindow	LONG	NO	
timestamp	LONG	YES	

`orderId` 与 `origClientOrderId` 必须至少发送一个

撤销全部订单 (TRADE)

响应:

```
{
    "code": "200",
    "msg": "The operation of cancel all open order is done."
}
```

`DELETE /fapi/v1/allOpenOrders (HMAC SHA256)`

权重: 1

Parameters:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
recvWindow	LONG	NO	

名称	类型	是否必需	描述
timestamp	LONG	YES	

批量撤销订单 (TRADE)

响应:

```
[
  {
    "clientOrderId": "myOrder1", // 用户自定义的订单号
    "cumQty": "0",
    "cumQuote": "0", // 成交金额
    "executedQty": "0", // 成交量
    "orderId": 283194212, // 系统订单号
    "origQty": "11", // 原始委托数量
    "price": "0", // 委托价格
    "reduceOnly": false, // 仅减仓
    "side": "BUY", // 买卖方向
    "positionSide": "SHORT", // 持仓方向
    "status": "CANCELED", // 订单状态
    "stopPrice": "9300", // 触发价, 对`TRAILING_STOP_MARKET`无效
    "closePosition": false, // 是否条件全平仓
    "symbol": "BTCUSDT", // 交易对
    "timeInForce": "GTC", // 有效方法
    "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
    "type": "TRAILING_STOP_MARKET", // 订单类型
    "activatePrice": "9020", // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET`订单返回此字段
    "priceRate": "0.3", // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET`订单返回此字段
    "updateTime": 1571110484038, // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false // 是否开启条件单触发保护
  },
  {
    "code": -2011,
    "msg": "Unknown order sent."
  }
]
```

`DELETE /fapi/v1/batchOrders (HMAC SHA256)`

权重: 1

Parameters:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
orderIdList	LIST<LONG>	NO	系统订单号, 最多支持10个订单 比如 [1234567, 2345678]
origClientOrderIdList	LIST<STRING>	NO	用户自定义的订单号, 最多支持10个订单 比如 ["my_id_1", "my_id_2"] 需要encode双引号。逗号后面没有空格。

名称	类型	是否必需	描述
recvWindow	LONG	NO	
timestamp	LONG	YES	

`orderIdList` 与 `origClientOrderIdList` 必须至少发送一个，不可同时发送

倒计时撤销所有订单 (TRADE)

响应:

```
{
  "symbol": "BTCUSDT",
  "countdownTime": "100000"
}
```

`POST /fapi/v1/countdownCancelAll (HMAC SHA256)`

权重: 10

Parameters:

名称	类型	是否必需	描述
symbol	STRING	YES	
countdownTime	LONG	YES	倒计时。 1000 表示 1 秒； 0 表示取消倒计时撤单功能。
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 该接口可以被用于确保在倒计时结束时撤销指定symbol上的所有挂单。在使用这个功能时，接口应像心跳一样在倒计时内被反复调用，以便可以取消既有的倒计时并开始新的倒数计时设置。
- 用法示例：以30s的间隔重复此接口，每次倒计时countdownTime设置为120000(120s)。
如果在120秒内未再次调用此接口，则您指定symbol上的所有挂单都会被自动撤销。
如果在120秒内以将countdownTime设置为0，则倒数计时器将终止，自动撤单功能取消。
- 系统会**大约每10毫秒**检查一次所有倒计时情况，因此请注意，使用此功能时应考虑足够的冗余。
我们不建议将倒计时设置得太精确或太小。

查询当前挂单 (USER_DATA)

响应:

```
{
  "avgPrice": "0.00000",           // 平均成交价
  "clientOrderId": "abc",          // 用户自定义的订单号
  "cumQuote": "0",                // 成交金额
}
```

```

    "executedQty": "0",           // 成交量
    "orderId": 1917641,          // 系统订单号
    "origQty": "0.40",           // 原始委托数量
    "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
    "price": "0",                // 委托价格
    "reduceOnly": false,         // 是否仅减仓
    "side": "BUY",              // 买卖方向
    "status": "NEW",             // 订单状态
    "positionSide": "SHORT",    // 持仓方向
    "stopPrice": "9300",         // 触发价, 对`TRAILING_STOP_MARKET`无效
    "closePosition": false,      // 是否条件全平仓
    "symbol": "BTCUSDT",         // 交易对
    "time": 1579276756075,       // 订单时间
    "timeInForce": "GTC",        // 有效方法
    "type": "TRAILING_STOP_MARKET", // 订单类型
    "activatePrice": "9020",     // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "priceRate": "0.3",          // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "updateTime": 1579276756075, // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false,        // 是否开启条件单触发保护
}

}

```

`GET /fapi/v1/openOrder (HMAC SHA256)`

请小心使用不带symbol参数的调用

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
orderId	LONG	NO	系统订单号
origClientOrderId	STRING	NO	用户自定义的订单号
recvWindow	LONG	NO	
timestamp	LONG	YES	

- `orderId` 与 `origClientOrderId` 中的一个为必填参数
- 查询的订单如果已经成交或取消，将回报错 "Order does not exist."

查看当前全部挂单 (USER_DATA)

响应:

```

[
{
    "avgPrice": "0.00000",           // 平均成交价
    "clientOrderId": "abc",          // 用户自定义的订单号
    "cumQuote": "0",                // 成交金额
    "executedQty": "0",             // 成交量
}

```

```

    "orderId": 1917641,           // 系统订单号
    "origQty": "0.40",           // 原始委托数量
    "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
    "price": "0",                // 委托价格
    "reduceOnly": false,          // 是否仅减仓
    "side": "BUY",               // 买卖方向
    "positionSide": "SHORT",     // 持仓方向
    "status": "NEW",              // 订单状态
    "stopPrice": "9300",          // 触发价, 对`TRAILING_STOP_MARKET` 无效
    "closePosition": false,       // 是否条件全平仓
    "symbol": "BTCUSDT",          // 交易对
    "time": 1579276756075,        // 订单时间
    "timeInForce": "GTC",          // 有效方法
    "type": "TRAILING_STOP_MARKET", // 订单类型
    "activatePrice": "9020",        // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "priceRate": "0.3",             // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "updateTime": 1579276756075,      // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false,           // 是否开启条件单触发保护
}
]

```

`GET /fapi/v1/openOrders (HMAC SHA256)`

请小心使用不带symbol参数的调用

权重: - 带symbol **1** - 不带 **40**

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 不带symbol参数, 会返回所有交易对的挂单

查询所有订单(包括历史订单) (USER_DATA)

响应:

```

[
{
    "avgPrice": "0.00000",           // 平均成交价
    "clientOrderId": "abc",          // 用户自定义的订单号
    "cumQuote": "0",                // 成交金额
    "executedQty": "0",              // 成交量
    "orderId": 1917641,              // 系统订单号
    "origQty": "0.40",               // 原始委托数量
    "origType": "TRAILING_STOP_MARKET", // 触发前订单类型
    "price": "0",                   // 委托价格
    "reduceOnly": false,              // 是否仅减仓
}
]

```

```

    "side": "BUY",                                // 买卖方向
    "positionSide": "SHORT", // 持仓方向
    "status": "NEW",                                // 订单状态
    "stopPrice": "9300",                            // 触发价, 对`TRAILING_STOP_MARKET` 无效
    "closePosition": false,                         // 是否条件全平仓
    "symbol": "BTCUSDT",                           // 交易对
    "time": 1579276756075,                         // 订单时间
    "timeInForce": "GTC",                           // 有效方法
    "type": "TRAILING_STOP_MARKET",                // 订单类型
    "activatePrice": "9020", // 跟踪止损激活价格, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "priceRate": "0.3", // 跟踪止损回调比例, 仅`TRAILING_STOP_MARKET` 订单返回此字段
    "updateTime": 1579276756075,                   // 更新时间
    "workingType": "CONTRACT_PRICE", // 条件价格触发类型
    "priceProtect": false,                          // 是否开启条件单触发保护
}
]

```

GET /fapi/v1/allOrders (HMAC SHA256)

- 请注意，如果订单满足如下条件，不会被查询到：
 - 订单的最终状态为 `CANCELED` 或者 `EXPIRED`, 并且
 - 订单没有任何的成交记录, 并且
 - 订单生成时间 + 7天 < 当前时间

权重: 5

Parameters:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
orderId	LONG	NO	只返回此orderID及之后的订单，缺省返回最近的订单
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	返回的结果集数量 默认值:500 最大值:1000
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 查询时间范围最大不得超过7天
- 默认查询最近7天内的数据

账户余额V2 (USER_DATA)

响应:

```

[
{
    "accountAlias": "SgsR", // 账户唯一识别码
    "asset": "USDT", // 资产
    "balance": "122607.35137903", // 总余额
}
]

```

```

    "crossWalletBalance": "23.72469206", // 全仓余额
    "crossUnPnl": "0.00000000" // 全仓持仓未实现盈亏
    "availableBalance": "23.72469206", // 下单可用余额
    "maxWithdrawAmount": "23.72469206", // 最大可转出余额
    "marginAvailable": true, // 是否可用作联合保证金
    "updateTime": 1617939110373
}
]

```

GET /fapi/v2/balance (HMAC SHA256)

Weight: 5

Parameters:

名称	类型	是否必需	描述
recvWindow	LONG	NO	
timestamp	LONG	YES	

账户信息V2 (USER_DATA)

响应:

单资产模式

```
{
    "feeTier": 0, // 手续费等级
    "canTrade": true, // 是否可以交易
    "canDeposit": true, // 是否可以入金
    "canWithdraw": true, // 是否可以出金
    "updateTime": 0, // 保留字段, 请忽略
    "totalInitialMargin": "0.00000000", // 当前所需起始保证金总额(存在逐仓请忽略), 仅计算usdt资产
    "totalMaintMargin": "0.00000000", // 维持保证金总额, 仅计算usdt资产
    "totalWalletBalance": "23.72469206", // 账户总余额, 仅计算usdt资产
    "totalUnrealizedProfit": "0.00000000", // 持仓未实现盈亏总额, 仅计算usdt资产
    "totalMarginBalance": "23.72469206", // 保证金总余额, 仅计算usdt资产
    "totalPositionInitialMargin": "0.00000000", // 持仓所需起始保证金(基于最新标记价格), 仅计算usdt资产
    "totalOpenOrderInitialMargin": "0.00000000", // 当前挂单所需起始保证金(基于最新标记价格), 仅计算usdt资产
    "totalCrossWalletBalance": "23.72469206", // 全仓账户余额, 仅计算usdt资产
    "totalCrossUnPnl": "0.00000000", // 全仓持仓未实现盈亏总额, 仅计算usdt资产
    "availableBalance": "23.72469206", // 可用余额, 仅计算usdt资产
    "maxWithdrawAmount": "23.72469206" // 最大可转出余额, 仅计算usdt资产
    "assets": [
        {
            "asset": "USDT", // 资产
            "walletBalance": "23.72469206", // 余额
            "unrealizedProfit": "0.00000000", // 未实现盈亏
            "marginBalance": "23.72469206", // 保证金余额
            "maintMargin": "0.00000000", // 维持保证金
            "initialMargin": "0.00000000", // 当前所需起始保证金
            "positionInitialMargin": "0.00000000", // 持仓所需起始保证金(基于最新标记价格)
            "openOrderInitialMargin": "0.00000000", // 当前挂单所需起始保证金(基于最新标记价格)
        }
    ]
}
```

```

    "crossWalletBalance": "23.72469206", //全仓账户余额
    "crossUnPnl": "0.00000000" // 全仓持仓未实现盈亏
    "availableBalance": "23.72469206", // 可用余额
    "maxWithdrawAmount": "23.72469206", // 最大可转出余额
    "marginAvailable": true, // 是否可用作联合保证金
    "updateTime": 1625474304765 //更新时间
},
{
    "asset": "BUSD", //资产
    "walletBalance": "103.12345678", //余额
    "unrealizedProfit": "0.00000000", // 未实现盈亏
    "marginBalance": "103.12345678", // 保证金余额
    "maintMargin": "0.00000000", // 维持保证金
    "initialMargin": "0.00000000", // 当前所需起始保证金
    "positionInitialMargin": "0.00000000", // 持仓所需起始保证金(基于最新标记价格)
    "openOrderInitialMargin": "0.00000000", // 当前挂单所需起始保证金(基于最新标记价格)
    "crossWalletBalance": "103.12345678", //全仓账户余额
    "crossUnPnl": "0.00000000" // 全仓持仓未实现盈亏
    "availableBalance": "103.12345678", // 可用余额
    "maxWithdrawAmount": "103.12345678", // 最大可转出余额
    "marginAvailable": true, // 否可用作联合保证金
    "updateTime": 0 // 更新时间
}
],
"positions": [ // 头寸, 将返回所有市场symbol。
    //根据用户持仓模式展示持仓方向, 即单向模式下只返回BOTH持仓情况, 双向模式下只返回 LONG 和 SHORT 持仓情况
{
    "symbol": "BTCUSDT", // 交易对
    "initialMargin": "0", // 当前所需起始保证金(基于最新标记价格)
    "maintMargin": "0", //维持保证金
    "unrealizedProfit": "0.00000000", // 持仓未实现盈亏
    "positionInitialMargin": "0", // 持仓所需起始保证金(基于最新标记价格)
    "openOrderInitialMargin": "0", // 当前挂单所需起始保证金(基于最新标记价格)
    "leverage": "100", // 杠杆倍率
    "isolated": true, // 是否是逐仓模式
    "entryPrice": "0.00000", // 持仓成本价
    "maxNotional": "250000", // 当前杠杆下用户可用的最大名义价值
    "bidNotional": "0", // 买单净值, 忽略
    "askNotional": "0", // 卖单净值, 忽略
    "positionSide": "BOTH", // 持仓方向
    "positionAmt": "0", // 持仓数量
    "updateTime": 0 // 更新时间
}
]
}

```

多资产模式

```
{
    "feeTier": 0, // 手续费等级
    "canTrade": true, // 是否可以交易
    "canDeposit": true, // 是否可以入金
    "canWithdraw": true, // 是否可以出金
    "updateTime": 0, // 保留字段, 请忽略
    "totalInitialMargin": "0.00000000", // 以USD计价的所需起始保证金总额
    "totalMaintMargin": "0.00000000", // 以USD计价的维持保证金总额
    "totalWalletBalance": "126.72469206", // 以USD计价的账户总余额
}
```

```

"totalUnrealizedProfit": "0.00000000", // 以USD计价的持仓未实现盈亏总额
"totalMarginBalance": "126.72469206", // 以USD计价的保证金总余额
"totalPositionInitialMargin": "0.00000000", // 以USD计价的持仓所需起始保证金(基于最新标记价格)
"totalOpenOrderInitialMargin": "0.00000000", // 以USD计价的当前挂单所需起始保证金(基于最新标记价格)
"totalCrossWalletBalance": "126.72469206", // 以USD计价的全仓账户余额
"totalCrossUnPnl": "0.00000000", // 以USD计价的全仓持仓未实现盈亏总额
"availableBalance": "126.72469206", // 以USD计价的可用余额
"maxWithdrawAmount": "126.72469206", // 以USD计价的最大可转出余额
"assets": [
    {
        "asset": "USDT", //资产
        "walletBalance": "23.72469206", //余额
        "unrealizedProfit": "0.00000000", // 未实现盈亏
        "marginBalance": "23.72469206", // 保证金余额
        "maintMargin": "0.00000000", // 维持保证金
        "initialMargin": "0.00000000", // 当前所需起始保证金
        "positionInitialMargin": "0.00000000", // 持仓所需起始保证金(基于最新标记价格)
        "openOrderInitialMargin": "0.00000000", // 当前挂单所需起始保证金(基于最新标记价格)
        "crossWalletBalance": "23.72469206", //全仓账户余额
        "crossUnPnl": "0.00000000" // 全仓持仓未实现盈亏
        "availableBalance": "23.72469206", // 可用余额
        "maxWithdrawAmount": "23.72469206", // 最大可转出余额
        "marginAvailable": true, // 是否可用作联合保证金
        "updateTime": 1625474304765 //更新时间
    },
    {
        "asset": "BUSD", //资产
        "walletBalance": "103.12345678", //余额
        "unrealizedProfit": "0.00000000", // 未实现盈亏
        "marginBalance": "103.12345678", // 保证金余额
        "maintMargin": "0.00000000", // 维持保证金
        "initialMargin": "0.00000000", // 当前所需起始保证金
        "positionInitialMargin": "0.00000000", // 持仓所需起始保证金(基于最新标记价格)
        "openOrderInitialMargin": "0.00000000", // 当前挂单所需起始保证金(基于最新标记价格)
        "crossWalletBalance": "103.12345678", //全仓账户余额
        "crossUnPnl": "0.00000000" // 全仓持仓未实现盈亏
        "availableBalance": "103.12345678", // 可用余额
        "maxWithdrawAmount": "103.12345678", // 最大可转出余额
        "marginAvailable": true, // 否可用作联合保证金
        "updateTime": 0 // 更新时间
    }
],
"positions": [ // 头寸, 将返回所有市场symbol。
    //根据用户持仓模式展示持仓方向, 即单向模式下只返回BOTH持仓情况, 双向模式下只返回 LONG 和 SHORT 持仓情况
    {
        "symbol": "BTCUSDT", // 交易对
        "initialMargin": "0", // 当前所需起始保证金(基于最新标记价格)
        "maintMargin": "0", //维持保证金
        "unrealizedProfit": "0.00000000", // 持仓未实现盈亏
        "positionInitialMargin": "0", // 持仓所需起始保证金(基于最新标记价格)
        "openOrderInitialMargin": "0", // 当前挂单所需起始保证金(基于最新标记价格)
        "leverage": "100", // 杠杆倍率
        "isolated": true, // 是否是逐仓模式
        "entryPrice": "0.00000", // 持仓成本价
        "maxNotional": "250000", // 当前杠杆下用户可用的最大名义价值
        "bidNotional": "0", // 买单净值, 忽略
        "askNotional": "0", // 卖单净值, 忽略
        "positionSide": "BOTH", // 持仓方向
    }
]
}

```

```

        "positionAmt": "0",      // 持仓数量
        "updateTime": 0          // 更新时间
    }
]
}

```

GET /fapi/v2/account (HMAC SHA256)

现有账户信息。 用户在单资产模式和多资产模式下会看到不同结果，响应部分的注释解释了两种模式下的不同。

权重: 5

参数:

名称	类型	是否必需	描述
recvWindow	LONG	NO	
timestamp	LONG	YES	

调整仓位杠杆 (TRADE)

响应:

```

{
    "leverage": 21, // 杠杆倍数
    "maxNotionalValue": "1000000", // 当前杠杆倍数下允许的最大名义价值
    "symbol": "BTCUSDT" // 交易对
}

```

POST /fapi/v1/leverage (HMAC SHA256)

调整用户在指定symbol合约的仓位杠杆。

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
leverage	INT	YES	目标杠杆倍数：1 到 125 整数
recvWindow	LONG	NO	
timestamp	LONG	YES	

变换逐全仓模式 (TRADE)

响应:

```
{
  "code": 200,
  "msg": "success"
}
```

`POST /fapi/v1/marginType (HMAC SHA256)`

变换用户在指定symbol合约上的保证金模式：逐仓或全仓。

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
marginType	ENUM	YES	保证金模式 ISOLATED(逐仓), CROSSED(全仓)
recvWindow	LONG	NO	
timestamp	LONG	YES	

调整逐仓保证金 (TRADE)

响应:

```
{
  "amount": 100.0,
  "code": 200,
  "msg": "Successfully modify position margin.",
  "type": 1
}
```

`POST /fapi/v1/positionMargin (HMAC SHA256)`

针对逐仓模式下的仓位，调整其逐仓保证金资金。

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
positionSide	ENUM	NO	持仓方向，单向持仓模式下非必填，默认且仅可填 BOTH ；在双向持仓模式下必填且仅可选择 LONG 或 SHORT
amount	DECIMAL	YES	保证金资金
type	INT	YES	调整方向 1: 增加逐仓保证金，2: 减少逐仓保证金
recvWindow	LONG	NO	

名称	类型	是否必需	描述
timestamp	LONG	YES	

- 只针对逐仓symbol 与 positionSide(如有)

逐仓保证金变动历史 (TRADE)

响应:

```
[
  {
    "amount": "23.36332311", // 数量
    "asset": "USDT", // 资产
    "symbol": "BTCUSDT", // 交易对
    "time": 1578047897183, // 时间
    "type": 1, // 调整方向
    "positionSide": "BOTH" // 持仓方向
  },
  {
    "amount": "100",
    "asset": "USDT",
    "symbol": "BTCUSDT",
    "time": 1578047900425,
    "type": 1,
    "positionSide": "LONG"
  }
]
```

`GET /fapi/v1/positionMargin/history (HMAC SHA256)`

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
type	INT	NO	调整方向 1: 增加逐仓保证金 , 2: 减少逐仓保证金
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	返回的结果集数量 默认值: 500
recvWindow	LONG	NO	
timestamp	LONG	YES	

用户持仓风险V2 (USER_DATA)

响应:

单向持仓模式下：

```
[
{
  "entryPrice": "0.00000", // 开仓均价
  "marginType": "isolated", // 逐仓模式或全仓模式
  "isAutoAddMargin": "false",
  "isolatedMargin": "0.00000000", // 逐仓保证金
  "leverage": "10", // 当前杠杆倍数
  "liquidationPrice": "0", // 参考强平价格
  "markPrice": "6679.50671178", // 当前标记价格
  "maxNotionalValue": "20000000", // 当前杠杆倍数允许的名义价值上限
  "positionAmt": "0.000", // 头寸数量, 符号代表多空方向, 正数为多, 负数为空
  "notional": "0",
  "isolatedWallet": "0",
  "symbol": "BTCUSDT", // 交易对
  "unRealizedProfit": "0.00000000", // 持仓未实现盈亏
  "positionSide": "BOTH", // 持仓方向
  "updateTime": 1625474304765 // 更新时间
},
]
```

双向持仓模式下：

```
[
{
  "symbol": "BTCUSDT", // 交易对
  "positionAmt": "0.001", // 头寸数量, 符号代表多空方向, 正数为多, 负数为空
  "entryPrice": "22185.2", // 开仓均价
  "markPrice": "21123.05052574", // 当前标记价格
  "unRealizedProfit": "-1.06214947", // 持仓未实现盈亏
  "liquidationPrice": "19731.45529116", // 参考强平价格
  "leverage": "4", // 当前杠杆倍数
  "maxNotionalValue": "100000000", // 当前杠杆倍数允许的名义价值上限
  "marginType": "cross", // 逐仓模式或全仓模式
  "isolatedMargin": "0.00000000", // 逐仓保证金
  "isAutoAddMargin": "false",
  "positionSide": "LONG", // 持仓方向
  "notional": "21.12305052",
  "isolatedWallet": "0",
  "updateTime": 1655217461579 // 更新时间
},
{
  "symbol": "BTCUSDT",
  "positionAmt": "0.000",
  "entryPrice": "0.0",
  "markPrice": "21123.05052574",
  "unRealizedProfit": "0.00000000",
  "liquidationPrice": "0",
  "leverage": "4",
  "maxNotionalValue": "100000000",
}
```

```

    "marginType": "cross",
    "isolatedMargin": "0.00000000",
    "isAutoAddMargin": "false",
    "positionSide": "SHORT",
    "notional": "0",
    "isolatedWallet": "0",
    "updateTime": 0
}
]

```

`GET /fapi/v2/positionRisk (HMAC SHA256)`

权重: 5

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	
recvWindow	LONG	NO	
timestamp	LONG	YES	

注意

请与账户推送信息`ACCOUNT_UPDATE`配合使用，以满足您的及时性和准确性需求。

账户成交历史 (USER_DATA)

响应:

```

[
{
    "buyer": false, // 是否是买方
    "commission": "-0.07819010", // 手续费
    "commissionAsset": "USDT", // 手续费计价单位
    "id": 698759, // 交易ID
    "maker": false, // 是否是挂单方
    "orderId": 25851813, // 订单编号
    "price": "7819.01", // 成交价
    "qty": "0.002", // 成交量
    "quoteQty": "15.63802", // 成交额
    "realizedPnl": "-0.91539999", // 实现盈亏
    "side": "SELL", // 买卖方向
    "positionSide": "SHORT", // 持仓方向
    "symbol": "BTCUSDT", // 交易对
    "time": 1569514978020 // 时间
}
]

```

`GET /fapi/v1/userTrades (HMAC SHA256)`

获取某交易对的成交历史

权重: 5

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	交易对
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
fromId	LONG	NO	返回该fromId及之后的成交，缺省返回最近的成交
limit	INT	NO	返回的结果集数量，默认值:500 最大值:1000.
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 如果 startTime 和 endTime 均未发送, 只会返回最近7天的数据。
- startTime 和 endTime 的最大间隔为7天

获取账户损益资金流水 (USER_DATA)

响应:

```
[
  {
    "symbol": "", // 交易对, 仅针对涉及交易对的资金流
    "incomeType": "TRANSFER", // 资金流类型
    "income": "-0.37500000", // 资金流数量, 正数代表流入, 负数代表流出
    "asset": "USDT", // 资产内容
    "info": "TRANSFER", // 备注信息, 取决于流水类型
    "time": 1570608000000, // 时间
    "tranId": "9689322392", // 划转ID
    "tradeId": "" // 引起流水产生的原始交易ID
  },
  {
    "symbol": "BTCUSDT",
    "incomeType": "COMMISSION",
    "income": "-0.01000000",
    "asset": "USDT",
    "info": "COMMISSION",
    "time": 1570636800000,
    "tranId": "9689322392",
    "tradeId": "2059192"
  }
]
```

GET /fapi/v1/income (HMAC SHA256)

权重: 30

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	交易对
incomeType	STRING	NO	收益类型： TRANSFER 转账, WELCOME_BONUS 欢迎奖金, REALIZED_PNL 已实现盈亏, FUNDING_FEE 差价金额, COMMISSION 佣金, INSURANCE_CLEAR 强平, REFERRAL_KICKBACK 推荐人返佣, COMMISSION_REBATE 被推荐人返佣, MARKET MAKER REBATE 手续费返还上账, API REBATE API佣金回扣, CONTEST_REWARD 交易大赛奖金, CROSS_COLLATERAL_TRANSFER cc转账, OPTIONS_PREMIUM_FEE 期权购置手续费, OPTIONS_SETTLE_PROFIT 期权行权收益, INTERNAL_TRANSFER 内部账户, 给普通用户划转, AUTO_EXCHANGE 自动兑换, DELIVERED_SETTELMENT 下架结算, COIN_SWAP_DEPOSIT 闪兑转入, COIN_SWAP_WITHDRAW 闪兑转出, POSITION_LIMIT_INCREASE_FEE 仓位限制上调费用
startTime	LONG	NO	起始时间
endTime	LONG	NO	结束时间
limit	INT	NO	返回的结果集数量 默认值:100 最大值:1000
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 如果 `startTime` 和 `endTime` 均未发送, 只会返回最近7天的数据。
- 如果 `incomeType` 没有发送, 返回所有类型账户损益资金流水。
- "trandId" 在相同用户的同一种收益流水类型中是唯一的。
- 仅保留最近3个月的数据。

杠杆分层标准 (USER_DATA)

响应:

```
[
  {
    "symbol": "ETHUSDT",
    "brackets": [
      {
        "bracket": 1, // 层级
        "initialLeverage": 75, // 该层允许的最高初始杠杆倍数
        "notionalCap": 10000, // 该层对应的名义价值上限
        "notionalFloor": 0, // 该层对应的名义价值下限
        "maintMarginRatio": 0.0065, // 该层对应的维持保证金率
        "cum": 0 // 速算数
      },
    ],
  }
]
```

或 (若发送symbol)

```
{
```

```

"symbol": "ETHUSDT",
"brackets": [
  {
    "bracket": 1,
    "initialLeverage": 75,
    "notionalCap": 10000,
    "notionalFloor": 0,
    "maintMarginRatio": 0.0065,
    "cum": 0
  },
]
}

```

`GET /fapi/v1/leverageBracket`

权重: 1

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	
recvWindow	LONG	NO	
timestamp	LONG	YES	

持仓ADL队列估算 (USER_DATA)

响应:

```

[
  {
    "symbol": "ETHUSDT",
    "adlQuantile": {
      // 对于全仓状态下的双向持仓模式的交易对，会返回 "LONG", "SHORT" 和 "HEDGE"，其中"HEDGE"的存在仅作指示出现，请忽略数值
      "LONG": 3,
      "SHORT": 3,
      "HEDGE": 0 // HEDGE 仅作为指示出现，请忽略数值
    }
  },
  {
    "symbol": "BTCUSDT",
    "adlQuantile": {
      // 对于单向持仓模式或者是逐仓状态下的双向持仓模式的交易对，会返回 "LONG", "SHORT" 和 "BOTH" 分别表示多头持仓的ADL队列估算分
      "LONG": 1, // 双开模式下多头持仓的ADL队列估算分
      "SHORT": 2, // 双开模式下空头持仓的ADL队列估算分
      "BOTH": 0 // 单开模式下持仓的ADL队列估算分
    }
  }
]

```

```
GET /fapi/v1/adlQuantile
```

权重: 5

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 每30秒更新数据
- 队列分数0 , 1 , 2 , 3 , 4 , 分数越高说明在ADL队列中的位置越靠前
- 对于单向持仓模式或者是逐仓状态下的双向持仓模式的交易对，会返回 "LONG", "SHORT" 和 "BOTH" 分别表示不同持仓方向上持仓的adl队列分数
- 对于全仓状态下的双向持仓模式的交易对，会返回 "LONG", "SHORT" 和 "HEDGE", 其中"HEDGE"的存在仅作为标记;其中如果多空均有持仓的情况下,"LONG"和"SHORT"返回共同计算后相同的队列分数。

用户强平单历史 (USER_DATA)

响应:

```
[
  {
    "orderId": 6071832819,
    "symbol": "BTCUSDT",
    "status": "FILLED",
    "clientOrderId": "autoclose-1596107620040000020",
    "price": "10871.09",
    "avgPrice": "10913.21000",
    "origQty": "0.001",
    "executedQty": "0.001",
    "cumQuote": "10.91321",
    "timeInForce": "IOC",
    "type": "LIMIT",
    "reduceOnly": false,
    "closePosition": false,
    "side": "SELL",
    "positionSide": "BOTH",
    "stopPrice": "0",
    "workingType": "CONTRACT_PRICE",
    "origType": "LIMIT",
    "time": 1596107620044,
    "updateTime": 1596107620087
  }
  [
    {
      "orderId": 6072734303,
      "symbol": "BTCUSDT",
      "status": "FILLED",
      "clientOrderId": "adl_autoclose",
      "time": 1596107620044
    }
  ]
]
```

```

    "price": "11023.14",
    "avgPrice": "10979.82000",
    "origQty": "0.001",
    "executedQty": "0.001",
    "cumQuote": "10.97982",
    "timeInForce": "GTC",
    "type": "LIMIT",
    "reduceOnly": false,
    "closePosition": false,
    "side": "BUY",
    "positionSide": "SHORT",
    "stopPrice": "0",
    "workingType": "CONTRACT_PRICE",
    "origType": "LIMIT",
    "time": 1596110725059,
    "updateTime": 1596110725071
}
]

```

`GET /fapi/v1/forceOrders`

权重: 带symbol 20, 不带symbol 50

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	
autoCloseType	ENUM	NO	"LIQUIDATION": 强平单, "ADL": ADL减仓单.
startTime	LONG	NO	
endTime	LONG	NO	
limit	INT	NO	Default 50; max 100.
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 如果没有传 "autoCloseType", 强平单和ADL减仓单都会被返回
- 如果没有传"startTime", 只会返回"endTime"之前7天内的数据

合约交易量化规则指标 (USER_DATA)

- 更多细节, 请参考合约交易量化规则

响应:

```
{
  "indicators": { // indicator:风控指标名, value:用户在该市场的风控指标数值, triggerValue:阈值, 对于没有达到记
    "BTCUSDT": [
      {
        "isLocked": true, // 用户该品种交易是否被风控禁用
        "plannedRecoverTime": 1545741270000, // 预计恢复时间, 若当前时间大于等于预计恢复时间则为空
      }
    ]
  }
}
```

```

    "indicator": "UFR", // Unfilled Ratio (UFR)
    "value": 0.05, // Current value
    "triggerValue": 0.995 // Trigger value
},
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "IFER", // IOC/FOK Expiration Ratio (IFER)
    "value": 0.99, // Current value
    "triggerValue": 0.99 // Trigger value
},
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "GCR", // GTC Cancellation Ratio (GCR)
    "value": 0.99, // Current value
    "triggerValue": 0.99 // Trigger value
},
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "DR", // Dust Ratio (DR)
    "value": 0.99, // Current value
    "triggerValue": 0.99 // Trigger value
}
],
"ETHUSDT": [
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "UFR",
    "value": 0.05,
    "triggerValue": 0.995
},
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "IFER",
    "value": 0.99,
    "triggerValue": 0.99
},
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "GCR",
    "value": 0.99,
    "triggerValue": 0.99
},
{
    "isLocked": true, // 用户该品种交易是否被风控禁用
    "plannedRecoverTime": 1545741270000,
    "indicator": "DR",
    "value": 0.99,
    "triggerValue": 0.99
}
]
},

```

```

    "updateTime": 1545741270000 // 返回值的更新时间
}

```

或(触发账号层级违规时)

```

{
  "indicators": [
    "ACCOUNT": [
      {
        "indicator": "TMV", // Too many violations 多交易对触发账号层级违规
        "value": 10,
        "triggerValue": 1,
        "plannedRecoverTime": 1644919865000,
        "isLocked": true
      }
    ]
  ],
  "updateTime": 1644913304748
}

```

`GET /fapi/v1/apiTradingStatus`

权重:

- 带 symbol **1**
- 不带 **10**

参数:

名称	类型	是否必需	描述
symbol	STRING	NO	
recvWindow	LONG	NO	
timestamp	LONG	YES	

用户手续费率 (USER_DATA)

响应:

```

{
  "symbol": "BTCUSDT",
  "makerCommissionRate": "0.0002", // 0.02%
  "takerCommissionRate": "0.0004" // 0.04%
}

```

`GET /fapi/v1/commissionRate (HMAC SHA256)`

权重: 20

参数:

名称	类型	是否必需	描述
symbol	STRING	YES	
recvWindow	LONG	NO	
timestamp	LONG	YES	

获取合约资金流水下载Id (USER_DATA)

响应:

```
{
    "avgCostTimestampOfLast30d": 7241837, //过去30天平均数据下载时间
    "downloadId": "546975389218332672", //下载Id
}
```

GET /fapi/v1/income/asyn (HMAC SHA256)

权重: 5

参数:

名称	类型	是否必需	描述
startTime	LONG	YES	起始时间
endTime	LONG	YES	结束时间
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 存在每月5次的请求限制，网页端和Rest接口下载次数共用。

通过下载Id获取合约资金流水下载链接 (USER_DATA)

响应:

```
{
    "downloadId": "545923594199212032", // 下载Id
    "status": "completed", // 状态，枚举类型: completed 已完成, processing 处理中
    "url": "www.binance.com", // 适配该笔ID请求的下载链接
    "notified": true, // 忽略
    "expirationTimestamp": 1645009771000, // 晚于该时间戳之后链接将自动失效
    "isExpired": null,
}
```

或 (服务器仍在处理中会返回)

```
{
  "downloadId": "545923594199212032",
  "status": "processing",
  "url": "",
  "notified": false,
  "expirationTimestamp": -1
  "isExpired": null,
}
```

GET /fapi/v1/income/asyn/id (HMAC SHA256)

权重: 5

参数:

名称	类型	是否必需	描述
downloadId	STRING	YES	通过下载id 接口获取
recvWindow	LONG	NO	
timestamp	LONG	YES	

- 下载链接有效期：24小时。

WebSocket 账户信息推送

- 本篇所列出REST接口的baseurl <https://fapi.binance.com>
- 用于订阅账户数据的 `listenKey` 从创建时刻起有效期为60分钟
- 可以通过 `PUT` 一个 `listenKey` 延长60分钟有效期
- 可以通过 `DELETE` 一个 `listenKey` 立即关闭当前数据流，并使该 `listenKey` 无效
- 在具有有效 `listenKey` 的帐户上执行 `POST` 将返回当前有效的 `listenKey` 并将其有效期延长60分钟
- 本篇所列出的websocket接口，共有如下两种连接方式：
 - 连接方式一：
 - Base Url: <wss://fstream.binance.com>
 - 订阅账户数据流的stream名称为 `/ws/<listenKey>`
 - 连接样例：
`wss://fstream.binance.com/ws/XaEAKTsQSRLZAGH9tuIu37p1SRsdjm1AVBoNYPUI1TAko1WI22PgmBMpI1rS8Yh`
 - 连接方式二：
 - Base Url: <wss://fstream-auth.binance.com>
 - 订阅账户数据流的stream名称为 `/ws/<listenKey>?listenKey=<validateListenKey>`
 - `<validateListenKey>` 在建立连接时，必须为一个有效的 `listenKey`
 - 连接样例：
`wss://fstream-auth.binance.com/ws/XaEAKTsQSRLZAGH9tuIu37p1SRsdjm1AVBoNYPUI1TAko1WI22PgmBMpI1rS8Yh? listenKey=XaEAKTsQSRLZAGH9tuIu37p1SRsdjm1AVBoNYPUI1TAko1WI22PgmBMpI1rS8Yh`
- 每个链接有效期不超过24小时，请妥善处理断线重连。
- 单一账户，单一连接的推送数据流消息可以保证时间序，**强烈建议您使用 E 字段进行排序**
- 考虑到剧烈行情下，RESTful接口可能存在查询延迟，我们强烈建议您优先从 WebSocket user data stream推送的消息来获取订单，仓位等信息。

生成listenKey (USER_STREAM)

响应:

```
{  
    "listenKey": "pqia91ma19a5s61cv6a81va65sdf19v8a65a1a5s61cv6a81va65sdf19v8a65a1"  
}
```

`POST /fapi/v1/listenKey`

创建一个新的user data stream，返回值为一个listenKey，即websocket订阅的stream名称。如果该帐户具有有效的listenKey，则将返回该listenKey并将其有效期延长60分钟。

权重: 1

参数:

None

延长listenKey有效期 (USER_STREAM)

响应:

```
{}
```

`PUT /fapi/v1/listenKey`

有效期延长至本次调用后60分钟

权重: 1

参数:

None

关闭listenKey (USER_STREAM)

响应:

```
{}
```

`DELETE /fapi/v1/listenKey`

关闭某账户数据流

权重: 1

参数:

None

listenKey 过期推送

Payload:

```
{
  'e': 'listenKeyExpired',      // 事件类型
  'E': 1576653824250          // 事件时间
}
```

当前连接使用的有效listenKey过期时，user data stream 将会推送此事件。

注意:

- 此事件与websocket连接中断没有必然联系
- 只有正在连接中的有效listenKey过期时才会收到此消息
- 收到此消息后user data stream将不再更新，直到用户使用新的有效的listenKey

追加保证金通知

Payload:

```
{
  "e": "MARGIN_CALL",        // 事件类型
  "E": 1587727187525,        // 事件时间
  "cw": "3.16812045",        // 除去逐仓仓位保证金的钱包余额，仅在全仓 margin call 情况下推送此字段
  "p": [
    {
      "s": "ETHUSDT",          // symbol
      "ps": "LONG",            // 持仓方向
      "pa": "1.327",           // 仓位
      "mt": "CROSSED",         // 保证金模式
      "iw": "0",                // 若为逐仓，仓位保证金
      "mp": "187.17127",        // 标记价格
      "up": "-1.166074",        // 未实现盈亏
      "mm": "1.614445"          // 持仓需要的维持保证金
    }
  ]
}
```

- 当用户持仓风险过高，会推送此消息。
- 此消息仅作为风险指导信息，不建议用于投资策略。
- 在大波动市场行情下,不排除此消息发出的同时用户仓位已被强平的可能。

Balance和Position更新推送

Payload:

```
{
  "e": "ACCOUNT_UPDATE", // 事件类型
  "E": 1564745798939, // 事件时间
  "T": 1564745798938 , // 撮合时间
  "a": [
    {
      "m": "ORDER", // 事件推出原因
      "B": [
        {
          "a": "USDT", // 资产名称
          "wb": "122624.12345678", // 钱包余额
          "cw": "100.12345678", // 除去逐仓仓位保证金的钱包余额
          "bc": "50.12345678" // 除去盈亏与交易手续费以外的钱包余额改变量
        },
        {
          "a": "BUSD",
          "wb": "1.00000000",
          "cw": "0.00000000",
          "bc": "-49.12345678"
        }
      ],
      "P": [
        {
          "s": "BTCUSDT", // 交易对
          "pa": "0", // 仓位
          "ep": "0.0000", // 入仓价格
          "cr": "200", // (费前)累计实现损益
          "up": "0", // 持仓未实现盈亏
          "mt": "isolated", // 保证金模式
          "iw": "0.00000000", // 若为逐仓，仓位保证金
          "ps": "BOTH" // 持仓方向
        },
        {
          "s": "BTCUSDT",
          "pa": "20",
          "ep": "6563.66500",
          "cr": "0",
          "up": "2850.21200",
          "mt": "isolated",
          "iw": "13200.70726908",
          "ps": "LONG"
        },
        {
          "s": "BTCUSDT",
          "pa": "-10",
          "ep": "6563.86000",
          "cr": "-45.04000000",
          "up": "-1423.15600",
          "mt": "isolated",
          "iw": "6570.42511771",
          "ps": "SHORT"
        }
      ]
    }
  ]
}
```

账户更新事件的 event type 固定为 **ACCOUNT_UPDATE**

- 当账户信息有变动时，会推送此事件：
 - 仅当账户信息有变动时(包括资金、仓位、保证金模式等发生变化)，才会推送此事件；
 - 订单状态变化没有引起账户和持仓变化的，不会推送此事件；
 - position 信息：仅当symbol仓位有变动时推送。
- "FUNDING FEE" 引起的资金余额变化，仅推送简略事件：
 - 当用户某全仓持仓发生"FUNDING FEE"时，事件 **ACCOUNT_UPDATE** 将只会推送相关的用户资产余额信息(B(仅推送 FUNDING FEE 发生相关的资产余额信息))，而不会推送任何持仓信息(P)。
 - 当用户某逐仓持仓发生"FUNDING FEE"时，事件 **ACCOUNT_UPDATE** 将只会推送相关的用户资产余额信息(B(仅推送 "FUNDING FEE" 所使用的资产余额信息))，和相关的持仓信息(P(仅推送这笔"FUNDING FEE"发生所在的持仓信息))，其余持仓信息不会被推送。
- 字段"m"代表了事件推出的原因，包含了以下可能类型：
 - DEPOSIT
 - WITHDRAW
 - ORDER
 - FUNDING_FEE
 - WITHDRAW_REJECT
 - ADJUSTMENT
 - INSURANCE_CLEAR
 - ADMIN_DEPOSIT
 - ADMIN_WITHDRAW
 - MARGIN_TRANSFER
 - MARGIN_TYPE_CHANGE
 - ASSET_TRANSFER
 - OPTIONS_PREMIUM_FEE
 - OPTIONS_SETTLE_PROFIT
 - AUTO_EXCHANGE
- 字段"bc"代表了钱包余额的改变量，即 balance change，但注意其不包含仓位盈亏及交易手续费。

订单/交易 更新推送

Payload:

```
{
  "e": "ORDER_TRADE_UPDATE",           // 事件类型
  "E": 1568879465651,                  // 事件时间
  "T": 1568879465650,                  // 撮合时间
  "o": {
    "s": "BTCUSDT",                    // 交易对
    "c": "TEST",                      // 客户端自定订单ID
    "S": "SELL",                      // 特殊的自定义订单ID:
    "o": "TRAILING_STOP_MARKET",     // 订单方向
    "f": "GTC",                        // 订单类型
    "q": "0.001",                      // 有效方式
    "p": "0",                          // 订单原始数量
    "ap": "0",                         // 订单原始价格
    "sp": "7103.04",                   // 订单平均价格
    "l": "0.001",                      // 条件订单触发价格，对追踪止损单无效
  }
}
```

```

    "x": "NEW",                                // 本次事件的具体执行类型
    "X": "NEW",                                // 订单的当前状态
    "i": 8886774,                             // 订单ID
    "l": "0",                                   // 订单末次成交量
    "z": "0",                                   // 订单累计已成交量
    "L": "0",                                   // 订单末次成交价格
    "N": "USDT",                               // 手续费资产类型
    "n": "0",                                   // 手续费数量
    "T": 1568879465650,                         // 成交时间
    "t": 0,                                     // 成交ID
    "b": "0",                                   // 买单净值
    "a": "9.91",                               // 卖单净值
    "m": false,                                // 该成交是作为挂单成交吗?
    "R": false,                                // 是否是只减仓单
    "wt": "CONTRACT_PRICE",                   // 触发价类型
    "ot": "TRAILING_STOP_MARKET",             // 原始订单类型
    "ps": "LONG",                              // 持仓方向
    "cp": false,                               // 是否为触发平仓单; 仅在条件订单情况下会推送此字段
    "AP": "7476.89",                           // 追踪止损激活价格, 仅在追踪止损单时会推送此字段
    "cr": "5.0",                               // 追踪止损回调比例, 仅在追踪止损单时会推送此字段
    "pP": false,                               // 忽略
    "si": 0,                                    // 忽略
    "ss": 0,                                    // 忽略
    "rp": "0"                                   // 该交易实现盈亏
}
}

```

当有新订单创建、订单有新成交或者新的状态变化时会推送此类事件 事件类型统一为 `ORDER_TRADE_UPDATE`

订单方向

- BUY 买入
- SELL 卖出

订单类型

- MARKET 市价单
- LIMIT 限价单
- STOP 止损单
- TAKE_PROFIT 止盈单
- LIQUIDATION 强平单

本次事件的具体执行类型

- NEW
- CANCELED 已撤
- CALCULATED
- EXPIRED 订单失效
- TRADE 交易

订单状态

- NEW
- PARTIALLY_FILLED
- FILLED
- CANCELED
- EXPIRED
- NEW_INSURANCE 风险保障基金(强平)
- NEW_ADL 自动减仓序列(强平)

有效方式:

- GTC
- IOC
- FOK
- GTX

强平和ADL:

- 若用户因保证金不足发生强平
 - 强平对手方为市场 : c 为 "autoclose-XXX" , X 为 "NEW"
 - 强平对手方为保险基金 : c 为 "autoclose-XXX" , X 为 "NEW_INSURANCE"
 - 强平对手方为ADL对手方 : c 为 "autoclose-XXX" , X 为 "NEW_ADL"
- 若用户保证金充足但被ADL:
 - c 为 "adl_autoclose" , X 为 "NEW"

杠杆倍数等账户配置 更新推送

Payload:

```
{
  "e": "ACCOUNT_CONFIG_UPDATE",           // 事件类型
  "E": 1611646737479,                    // 事件时间
  "T": 1611646737476,                    // 撮合时间
  "ac": {
    "s": "BTCUSDT",                      // 交易对
    "l": 25                             // 杠杆倍数
  }
}
```

Or

```
{
  "e": "ACCOUNT_CONFIG_UPDATE",           // 事件类型
  "E": 1611646737479,                    // 事件时间
  "T": 1611646737476,                    // 撮合时间
  "ai": {                                // 用户账户配置
    "j": true                           // 联合保证金状态
  }
}
```

当账户配置发生变化时会推送此类事件类型统一为 **ACCOUNT_CONFIG_UPDATE**

当交易对杠杆倍数发生变化时推送消息体会包含对象 ac 表示交易对账户配置，其中 s 代表具体的交易对， l 代表杠杆倍数

当用户联合保证金状态发生变化时推送消息体会包含对象 ai 表示用户账户配置，其中 j 代表用户联合保证金状态

统一账户接口

为了给币安合约用户提供更加优质的服务及提高用户的资金利用率，币安将推出统一账户计划。该计划将以合约钱包、现货杠杆钱包的总资产作为保证金来计算。关于统一帐户：币安统一帐户计划是一项跨资产保证金计划，支持超过 200 种有效的加密资产。U本位合约、币本位合约以及杠杆钱包中支持的加密资产和头寸将作为有效的联合抵押品，以确定统一账户的权益、保证金余额和维持保证金要求。

FAQ: 币安合约统一账户总览

仅对特定用户开放此功能，详情：加入统一账户计划

获取统一账户交易规则

响应：

```
{
  "notionalLimits": [ // 统一账户持仓上限
    {
      "symbol": "BTCUSDT", // 交易对
      "notionalLimit": "100000000" // 统一账户持仓上限USDT价值
    },
    {
      "symbol": "ETHUSDT",
      "notionalLimit": "20000000"
    },
  ]
}
```

`GET /fapi/v1/pmExchangeInfo`

获取统一账户交易规则

参数：

名称	类型	是否必需	描述
symbol	STRING	NO	交易对

查询统一账户账户信息 (USER_DATA)

响应：

```
{
  "maxWithdrawAmountUSD": "1627523.32459208", //统一账户以USD计价的最大可转出余额
  "asset": "BTC", // 资产
  "maxWithdrawAmount": "27.43689636", //最大可转出余额
}
```

`GET /fapi/v1/pmAccountInfo`

查询统一账户当前账户信息

权重(IP)： 5

参数：

名称	类型	是否必需	描述
asset	STRING	YES	
recvWindow	LONG	NO	

- 最大可转出余额指可以转出到现货钱包到金额。

错误代码

error JSON payload:

```
{
  "code": -1121,
  "msg": "Invalid symbol."
}
```

错误由两部分组成：错误代码和消息。 代码是通用的，但是消息可能会有所不同。

10xx - 常规服务器或网络问题

-1000 UNKNOWN

- An unknown error occurred while processing the request.
- 处理请求时发生未知错误。

-1001 DISCONNECTED

- Internal error; unable to process your request. Please try again.
- 内部错误；无法处理您的请求。请再试一次。

-1002 UNAUTHORIZED

- You are not authorized to execute this request.
- 您无权执行此请求。

-1003 TOO_MANY_REQUESTS

- Too many requests queued.
- 排队的请求过多。
- Too many requests; please use the websocket for live updates.
- 请求权重过多；请使用websocket获取最新更新。
- Too many requests; current limit is %s requests per minute. Please use the websocket for live updates to avoid polling the API.
- 请求权重过多；当前限制为每分钟%s请求权重。请使用websocket进行实时更新，以避免轮询API。
- Way too many requests; IP banned until %s. Please use the websocket for live updates to avoid bans.
- 请求权重过多；IP被禁止，直到%s。请使用websocket进行实时更新，以免被禁。

-1004 DUPLICATE_IP

- This IP is already on the white list
- IP地址已经在白名单

-1005 NO SUCH_IP

- No such IP has been white listed
• 白名单上没有此IP地址

-1006 UNEXPECTED_RESP

- An unexpected response was received from the message bus. Execution status unknown.
• 从消息总线收到意外的响应。执行状态未知。

-1007 TIMEOUT

- Timeout waiting for response from backend server. Send status unknown; execution status unknown.
• 等待后端服务器响应超时。发送状态未知；执行状态未知。

-1014 UNKNOWN_ORDER_COMPOSITION

- Unsupported order combination.
• 不支持当前的下单参数组合

-1015 TOO_MANY_ORDERS

- Too many new orders.
• 新订单太多。
• Too many new orders; current limit is %s orders per %. * 新订单太多；当前限制为每%s %s个订单。

-1016 SERVICE_SHUTTING_DOWN

- This service is no longer available.
• 该服务不可用。

-1020 UNSUPPORTED_OPERATION

- This operation is not supported.
• 不支持此操作。

-1021 INVALID_TIMESTAMP

- Timestamp for this request is outside of the recvWindow.
◦ 此请求的时间戳在recvWindow之外。
• Timestamp for this request was 1000ms ahead of the server's time. * 此请求的时间戳比服务器时间提前1000毫秒。

-1022 INVALID_SIGNATURE

- Signature for this request is not valid.
• 此请求的签名无效。

-1023 START_TIME_GREATER_THAN_END_TIME

- Start time is greater than end time.
• 参数里面的开始时间在结束时间之后

11xx - Request issues

-1100 ILLEGAL_CHARS

- Illegal characters found in a parameter.
• 在参数中发现非法字符。
• Illegal characters found in parameter '%s'; legal range is '%s'.
• 在参数 %s 中发现非法字符；合法范围是 %s。

-1101 TOO_MANY_PARAMETERS

- Too many parameters sent for this endpoint.
• 为此端点发送的参数太多。
• Too many parameters; expected '%s' and received '%s'.
• 参数太多；预期为 %s 并收到了 %s。

- Duplicate values for a parameter detected. * 检测到的参数值重复。

-1102 MANDATORY_PARAM_EMPTY_OR_MALFORMED

- A mandatory parameter was not sent, was empty/null, or malformed.
- 未发送强制性参数，该参数为空/空或格式错误。
- Mandatory parameter '%s' was not sent, was empty/null, or malformed. * 强制参数 %s 未发送，为空/空或格式错误。
- Param '%s' or '%s' must be sent, but both were empty/null! * 必须发送参数 %s 或 %s，但两者均为空！

-1103 UNKNOWN_PARAM

- An unknown parameter was sent.
- 发送了未知参数。

-1104 UNREAD_PARAMETERS

- Not all sent parameters were read.
- 并非所有发送的参数都被读取。
- Not all sent parameters were read; read '%s' parameter(s) but was sent '%s'.
- 并非所有发送的参数都被读取；读取了 %s 参数，但被发送了 %s。

-1105 PARAM_EMPTY

- A parameter was empty.
- 参数为空。
- Parameter '%s' was empty.
- 参数 %s 为空。

-1106 PARAM_NOT_REQUIRED

- A parameter was sent when not required.
- 发送了不需要的参数。
- Parameter '%s' sent when not required.
- 发送了不需要参数 %s。

-1111 BAD_PRECISION

- Precision is over the maximum defined for this asset.
- 精度超过为此资产定义的最大值。

-1112 NO_DEPTH

- No orders on book for symbol.
- 交易对没有挂单。

-1114 TIF_NOT_REQUIRED

- TimeInForce parameter sent when not required.
- 发送的 TimeInForce 参数不需要。

-1115 INVALID_TIF

- Invalid timeInForce.
- 无效的 timeInForce

-1116 INVALID_ORDER_TYPE

- Invalid orderType.
- 无效订单类型。

-1117 INVALID_SIDE

- Invalid side.
- 无效买卖方向。

-1118 EMPTY_NEW_CL_ORD_ID

- New client order ID was empty.

- 新的客户订单ID为空。

-1119 EMPTY_ORG_CL_ORD_ID

- Original client order ID was empty.
- 客户自定义的订单ID为空。

-1120 BAD_INTERVAL

- Invalid interval.
- 无效时间间隔。

-1121 BAD_SYMBOL

- Invalid symbol.
- 无效的交易对。

-1125 INVALID_LISTEN_KEY

- This `listenKey` does not exist.
- 此`listenKey`不存在。

-1127 MORE_THAN_XX_HOURS

- Lookup interval is too big.
- 查询间隔太大。
- More than %s hours between startTime and endTime.
- 从开始时间到结束时间之间超过%s小时。

-1128 OPTIONAL_PARAMS_BAD_COMBO

- Combination of optional parameters invalid.
- 可选参数组合无效。

-1130 INVALID_PARAMETER

- Invalid data sent for a parameter.
- 发送的参数为无效数据。
- Data sent for parameter '%s' is not valid.
- 发送参数%s的数据无效。

-1136 INVALID_NEW_ORDER_RESP_TYPE

- Invalid newOrderRespType.
- 无效的 newOrderRespType。

20xx - Processing Issues

-2010 NEW_ORDER_REJECTED

- `NEW_ORDER_REJECTED`
- 新订单被拒绝

-2011 CANCEL_REJECTED

- `CANCEL_REJECTED`
- 取消订单被拒绝

-2013 NO SUCH ORDER

- Order does not exist.
- 订单不存在。

-2014 BAD_API_KEY_FMT

- API-key format invalid.
- API-key 格式无效。

-2015 REJECTED_MBX_KEY

- Invalid API-key, IP, or permissions for action.
- 无效的API密钥，IP或操作权限。

-2016 NO_TRADING_WINDOW

- No trading window could be found for the symbol. Try ticker/24hrs instead.
- 找不到该交易对的交易窗口。尝试改为24小时自动报价。

-2018 BALANCE_NOT_SUFFICIENT

- Balance is insufficient.
- 余额不足

-2019 MARGIN_NOT_SUFFICIENT

- Margin is insufficient.
- 杠杆账户余额不足

-2020 UNABLE_TO_FILL

- Unable to fill.
- 无法成交

-2021 ORDER_WOULD_IMMEDIATELY_TRIGGER

- Order would immediately trigger.
- 订单可能被立刻触发

-2022 REDUCE_ONLY_REJECT

- ReduceOnly Order is rejected.
- **ReduceOnly** 订单被拒绝

-2023 USER_IN_LIQUIDATION

- User in liquidation mode now.
- 用户正处于被强平模式

-2024 POSITION_NOT_SUFFICIENT

- Position is not sufficient.
- 持仓不足

-2025 MAX_OPEN_ORDER_EXCEEDED

- Reach max open order limit.
- 挂单量达到上限

-2026 REDUCE_ONLY_ORDER_TYPE_NOT_SUPPORTED

- This OrderType is not supported when reduceOnly.
- 当前订单类型不支持 **reduceOnly**

-2027 MAX_LEVERAGE_RATIO

- Exceeded the maximum allowable position at current leverage.
- 挂单或持仓超出当前初始杠杆下的最大值

-2028 MIN_LEVERAGE_RATIO

- Leverage is smaller than permitted: insufficient margin balance.
- 调整初始杠杆过低，导致可用余额不足

40xx - Filters and other Issues

-4000 INVALID_ORDER_STATUS

- Invalid order status.
- 订单状态不正确

-4001 PRICE_LESS_THAN_ZERO

- Price less than 0.
- 价格小于0

-4002 PRICE_GREATER_THAN_MAX_PRICE

- Price greater than max price.
- 价格超过最大值

-4003 QTY_LESS_THAN_ZERO

- Quantity less than zero.
- 数量小于0

-4004 QTY_LESS_THAN_MIN_QTY

- Quantity less than min quantity.
- 数量小于最小值

-4005 QTY_GREATER_THAN_MAX_QTY

- Quantity greater than max quantity.
- 数量大于最大值

-4006 STOP_PRICE_LESS_THAN_ZERO

- Stop price less than zero.
- 触发价小于最小值

-4007 STOP_PRICE_GREATER_THAN_MAX_PRICE

- Stop price greater than max price.
- 触发价大于最大值

-4008 TICK_SIZE_LESS_THAN_ZERO

- Tick size less than zero.
- 价格精度小于0

-4009 MAX_PRICE_LESS_THAN_MIN_PRICE

- Max price less than min price.
- 最大价格小于最小价格

-4010 MAX_QTY_LESS_THAN_MIN_QTY

- Max qty less than min qty.
- 最大数量小于最小数量

-4011 STEP_SIZE_LESS_THAN_ZERO

- Step size less than zero.
- 步进值小于0

-4012 MAX_NUM_ORDERS_LESS_THAN_ZERO

- Max num orders less than zero.
- 最大订单量小于0

-4013 PRICE_LESS_THAN_MIN_PRICE

- Price less than min price.
- 价格小于最小价格

-4014 PRICE_NOT_INCREASED_BY_TICK_SIZE

- Price not increased by tick size.
- 价格增量不是价格精度的倍数。

-4015 INVALID_CL_ORD_ID_LEN

- Client order id is not valid.
- 客户订单ID有误。
- Client order id length should not be more than 36 chars
- 客户订单ID长度应该不多于36字符

-4016 PRICE_HIGHER_THAN_MULTIPLIER_UP

- Price is higher than mark price multiplier cap.

-4017 MULTIPLIER_UP_LESS_THAN_ZERO

- Multiplier up less than zero.
- 价格上限小于0

-4018 MULTIPLIER_DOWN_LESS_THAN_ZERO

- Multiplier down less than zero.
- 价格下限小于0

-4019 COMPOSITE_SCALE_OVERFLOW

- Composite scale too large.

-4020 TARGET_STRATEGY_INVALID

- Target strategy invalid for orderType '%s', reduceOnly '%b'.
- 目标策略值不适合 %s 订单状态, 只减仓 %b。

-4021 INVALID_DEPTH_LIMIT

- Invalid depth limit.
- 深度信息的 limit 值不正确。
- '%s' is not valid depth limit.
- %s 不是合理的深度信息的 limit 值。

-4022 WRONG_MARKET_STATUS

- market status sent is not valid.
- 发送的市场状态不正确。

-4023 QTY_NOT_INCREASED_BY_STEP_SIZE

- Qty not increased by step size.
- 数量的递增值不是步进值的倍数。

-4024 PRICE_LOWER_THAN_MULTIPLIER_DOWN

- Price is lower than mark price multiplier floor.

-4025 MULTIPLIER_DECIMAL_LESS_THAN_ZERO

- Multiplier decimal less than zero.

-4026 COMMISSION_INVALID

- Commission invalid.
- 收益值不正确

- `%s` less than zero.
- `%s` 少于0
- `%s` absolute value greater than `%s`
- `%s` 绝对值大于`%s`

-4027 INVALID_ACCOUNT_TYPE

- Invalid account type.
- 账户类型不正确。

-4028 INVALID_LEVERAGE

- Invalid leverage
- 杠杆倍数不正确
- Leverage `%s` is not valid
- 杠杆`%s`不正确
- Leverage `%s` already exist with `%s`
- 杠杆`%s`已经存在于`%s`

-4029 INVALID_TICK_SIZE_PRECISION

- Tick size precision is invalid.
- 价格精度小数点位数不正确。

-4030 INVALID_STEP_SIZE_PRECISION

- Step size precision is invalid.
- 步进值小数点位数不正确。

-4031 INVALID_WORKING_TYPE

- Invalid parameter working type
- 不正确的参数类型
- Invalid parameter working type: `%s`
- 不正确的参数类型: `%s`

-4032 EXCEED_MAX_CANCEL_ORDER_SIZE

- Exceed maximum cancel order size.
- 超过可以取消的最大订单量。
- Invalid parameter working type: `%s`
- 不正确的参数类型: `%s`

-4033 INSURANCE_ACCOUNT_NOT_FOUND

- Insurance account not found.
- 风险保障基金账号没找到。

-4044 INVALID_BALANCE_TYPE

- Balance Type is invalid.
- 余额类型不正确。

-4045 MAX_STOP_ORDER_EXCEEDED

- Reach max stop order limit.
- 达到止损单的上限。

-4046 NO_NEED_TO_CHANGE_MARGIN_TYPE

- No need to change margin type.
- 不需要切换仓位模式。

-4047 THERE_EXISTS_OPEN_ORDERS

- Margin type cannot be changed if there exists open orders.
- 如果有挂单，仓位模式不能切换。

-4048 THERE_EXISTS_QUANTITY

- Margin type cannot be changed if there exists position.
- 如果有仓位，仓位模式不能切换。

-4049 ADD_ISOLATED_MARGIN_REJECT

- Add margin only support for isolated position.

-4050 CROSS_BALANCE_INSUFFICIENT

- Cross balance insufficient.
- 全仓余额不足。

-4051 ISOLATED_BALANCE_INSUFFICIENT

- Isolated balance insufficient.
- 逐仓余额不足。

-4052 NO_NEED_TO_CHANGE_AUTO_ADD_MARGIN

- No need to change auto add margin.

-4053 AUTO_ADD_CROSSED_MARGIN_REJECT

- Auto add margin only support for isolated position.
- 自动增加保证金只适用于逐仓。

-4054 ADD_ISOLATED_MARGIN_NO_POSITION_REJECT

- Cannot add position margin: position is 0.
- 不能增加逐仓保证金: 持仓为0

-4055 AMOUNT_MUST_BE_POSITIVE

- Amount must be positive.
- 数量必须是正整数

-4056 INVALID_API_KEY_TYPE

- Invalid api key type.
- API key的类型不正确

-4057 INVALID_RSA_PUBLIC_KEY

- Invalid api public key
- API key不正确

-4058 MAX_PRICE_TOO_LARGE

- maxPrice and priceDecimal too large,please check.
- maxPrice和priceDecimal太大，请检查。

-4059 NO_NEED_TO_CHANGE_POSITION_SIDE

- No need to change position side.
- 无需变更仓位方向

-4060 INVALID_POSITION_SIDE

- Invalid position side.
- 仓位方向不正确。

-4061 POSITION_SIDE_NOT_MATCH

- Order's position side does not match user's setting.
- 订单的持仓方向和用户设置不一致。

-4062 REDUCE_ONLY_CONFLICT

- Invalid or improper reduceOnly value.
- 仅减仓的设置不正确。

-4063 INVALID_OPTIONS_REQUEST_TYPE

- Invalid options request type
- 无效的期权请求类型

-4064 INVALID_OPTIONS_TIME_FRAME

- Invalid options time frame
- 无效的期权时间窗口

-4065 INVALID_OPTIONS_AMOUNT

- Invalid options amount
- 无效的期权数量

-4066 INVALID_OPTIONS_EVENT_TYPE

- Invalid options event type
- 无效的期权事件类型

-4067 POSITION_SIDE_CHANGE_EXISTS_OPEN_ORDERS

- Position side cannot be changed if there exists open orders.
- 如果有挂单，无法修改仓位方向。

-4068 POSITION_SIDE_CHANGE_EXISTS_QUANTITY

- Position side cannot be changed if there exists position.
- 如果有仓位，无法修改仓位方向。

-4069 INVALID_OPTIONS_PREMIUM_FEE

- Invalid options premium fee
- 无效的期权费

-4070 INVALID_CL_OPTIONS_ID_LEN

- Client options id is not valid.
- 客户的期权ID不合法
- Client options id length should be less than 32 chars
- 客户的期权ID长度应该小于32个字符

-4071 INVALID_OPTIONS_DIRECTION

- Invalid options direction
- 期权的方向无效

-4072 OPTIONS_PREMIUM_NOT_UPDATE

- premium fee is not updated, reject order
- 期权费没有更新

-4073 OPTIONS_PREMIUM_INPUT_LESS_THAN_ZERO

- input premium fee is less than 0, reject order
- 输入的期权费小于0

-4074 OPTIONS_AMOUNT_BIGGER_THAN_UPPER

- Order amount is bigger than upper boundary or less than 0, reject order

-4075 OPTIONS_PREMIUM_OUTPUT_ZERO

- output premium fee is less than 0, reject order

-4076 OPTIONS_PREMIUM_TOO_DIFF

- original fee is too much higher than last fee
- 期权的费用比之前的费用高

-4077 OPTIONS_PREMIUM_REACH_LIMIT

- place order amount has reached to limit, reject order
- 下单的数量达到上限

-4078 OPTIONS_COMMON_ERROR

- options internal error
- 期权内部系统错误

-4079 INVALID_OPTIONS_ID

- invalid options id
- invalid options id: %s
- duplicate options id %d for user %d
- 期权ID无效

-4080 OPTIONS_USER_NOT_FOUND

- user not found
- user not found with id: %s
- 用户找不到

-4081 OPTIONS_NOT_FOUND

- options not found
- options not found with id: %s
- 期权找不到

-4082 INVALID_BATCH_PLACE_ORDER_SIZE

- Invalid number of batch place orders.
- Invalid number of batch place orders: %s
- 批量下单的数量不正确

-4083 PLACE_BATCH_ORDERS_FAIL

- Fail to place batch orders.
- 无法批量下单

-4084 UPCOMING_METHOD

- Method is not allowed currently. Upcoming soon.
- 方法不支持

-4085 INVALID_NOTIONAL_LIMIT_COEF

- Invalid notional limit coefficient
- 期权的有限系数不正确

-4086 INVALID_PRICE_SPREAD_THRESHOLD

- Invalid price spread threshold
- 无效的价差阀值

-4087 REDUCE_ONLY_ORDER_PERMISSION

- User can only place reduce only order
- 用户只能下仅减仓订单

-4088 NO_PLACE_ORDER_PERMISSION

- User can not place order currently
- 用户当前不能下单

-4104 INVALID_CONTRACT_TYPE

- Invalid contract type
- 无效的合约类型

-4114 INVALID_CLIENT_TRAN_ID_LEN

- clientTranId is not valid
- clientTranId不正确
- Client tran id length should be less than 64 chars
- 客户的tranId长度应该小于64个字符

-4115 DUPLICATED_CLIENT_TRAN_ID

- clientTranId is duplicated
- clientTranId重复
- Client tran id should be unique within 7 days
- 客户的tranId应在7天内唯一

-4118 REDUCE_ONLY_MARGIN_CHECK_FAILED

- ReduceOnly Order Failed. Please check your existing position and open orders
- 仅减仓订单失败。请检查现有的持仓和挂单

-4131 MARKET_ORDER_REJECT

- The counterparty's best price does not meet the PERCENT_PRICE filter limit
- 交易对手的最高价格未达到PERCENT_PRICE过滤器限制

-4135 INVALID_ACTIVATION_PRICE

- Invalid activation price
- 无效的激活价格

-4137 QUANTITY_EXISTS_WITH_CLOSE_POSITION

- Quantity must be zero with closePosition equals true
- 数量必须为0，当closePosition为true时

-4138 REDUCE_ONLY_MUST_BE_TRUE

- Reduce only must be true with closePosition equals true
- Reduce only 必须为true，当closePosition为true时

-4139 ORDER_TYPE_CANNOT_BE_MKT

- Order type can not be market if it's unable to cancel
- 订单类型不能为市价单如果不能取消

-4140 INVALID_OPENING_POSITION_STATUS

- Invalid symbol status for opening position
- 无效的交易对状态

-4141 SYMBOL_ALREADY_CLOSED

- Symbol is closed
- 交易对已下架

-4142 STRATEGY_INVALID_TRIGGER_PRICE

- REJECT: take profit or stop order will be triggered immediately
- 拒绝：止盈止损单将立即被触发

-4144 INVALID_PAIR

- Invalid pair
- 无效的pair

-4161 ISOLATED_LEVERAGE_REJECT_WITH_POSITION

- Leverage reduction is not supported in Isolated Margin Mode with open positions
- 逐仓仓位模式下无法降低杠杆

-4164 MIN_NOTIONAL

- Order's notional must be no smaller than 5.0 (unless you choose reduce only)
- 订单的名义价值不可以小于5，除了使用reduce only
- Order's notional must be no smaller than %s (unless you choose reduce only)
- 订单的名义价值不可以小于 %s，除了使用reduce only

-4165 INVALID_TIME_INTERVAL

- Invalid time interval
- 无效的间隔
- Maximum time interval is %s days
- 最大的时间间隔为 %s 天

-4183 PRICE_HIGHER_THAN_STOP_MULTIPLIER_UP

- Price is higher than stop price multiplier cap.
- 止盈止损订单价格不应高于触发价与报价乘数上限的乘积
- Limit price can't be higher than %s.
- 止盈止损订单价格不应高于 %s

-4184 PRICE_LOWER_THAN_STOP_MULTIPLIER_DOWN

- Price is lower than stop price multiplier floor.
- 止盈止损订单价格不应低于触发价与报价乘数下限的乘积
- Limit price can't be lower than %s.
- 止盈止损订单价格不应低于 %s