Liqui public API

This api provides access to such information as tickers of currency pairs, active orders on different pairs, the latest trades for each pair etc.

All API requests are made from this address: https://api.liqui.io/api/3/<method name>/<pair listing> (https://api.liqui.io/api/3/<method name>/<pair listing>)

Currency pairs are hyphen-separated (-), e.g.: https://api.liqui.io/api/3/ticker/eth_btc-ltc_btc (https://api.liqui.io/api/3/ticker/eth_btc-ltc_btc)

You can use as many pairs in the listing as you wish. Duplicates are not allowed. It is also possible to use only one pair: https://api.liqui.io/api/3/ticker/eth_btc (https://api.liqui.io/api/3/ticker/eth_btc). A set of pairs works with all the methods presented in the public api except info. All information is cached every 2 seconds, so there's no point in making more frequent requests. All API responses have the following format JSON.

Important! API will display an error if we disable one of the pairs listed in your request. If you are not going to synchronize the state of pairs using the method info, you need to send the GET-parameter ignore_invalid equal to 1, e.g.: https://api.liqui.io/api/3/ticker/eth_btc-ltc_btc?ignore_invalid=1 (https://api.liqui.io/api/3/ticker/eth_btc-ltc_btc?ignore_invalid=1). Without the parameter ignore_invalid this request would have caused an error because of a non-existent pair.

Method info

This method provides all the information about currently active pairs, such as the maximum number of digits after the decimal point, the minimum price, the maximum price, the minimum transaction size, whether the pair is hidden, the commission for each pair.

Sample request: https://api.liqui.io/api/3/info (https://api.liqui.io/api/3/info)

Sample response:

```
{
    "server_time":1370814956,
    "pairs":{
        "eth_btc":{
            "decimal_places":3,
            "min_price":0.1,
            "max_price":400,
            "min_amount":0.01,
            "hidden":0,
            "fee":0.2
        }
        ...
    }
}
```

decimal_places: number of decimals allowed during trading.
min_price: minimum price allowed during trading.
max_price: maximum price allowed during trading.
min_amount: minimum sell / buy transaction size.
hidden: whether the pair is hidden, 0 or 1.
fee: commission for this pair.
A hidden pair (hidden=1) remains active but is not displayed in the list of pairs on the main page. The Commission is displayed for all users, it will not change even if it was reduced on your account in case of promotional pricing. If one of the pairs is disabled, it will simply disappear from the list.

Method ticker

This method provides all the information about currently active pairs, such as: the maximum price, the minimum price, average price, trade volume, trade volume in currency, the last trade, Buy and Sell price. All information is provided over the past 24 hours.

Sample request: https://api.liqui.io/api/3/ticker/eth_btc (https://api.liqui.io/api/3/ticker/eth_btc)

Sample response:

```
{
    "eth_btc":{
        "high":109.88,
        "low":91.14,
        "avg":100.51,
        "vol":1632898.2249,
        "vol_cur":16541.51969,
        "last":101.773,
        "buy":101.9,
        "sell":101.773,
        "updated":1370816308
    }
    ...
}
```

high: maximum price.
low: minimum price.
avg: average price.
vol: trade volume.
vol_cur: trade volume in currency.
last: the price of the last trade.
buy: buy price.
sell: sell price.
updated: last update of cache.

Method depth

This method provides the information about active orders on the pair. Additionally it accepts an optional GET-parameter limit, which indicates how many orders should be displayed (150 by default). Is set to less than 2000.

Sample request: https://api.liqui.io/api/3/depth/eth_btc (https://api.liqui.io/api/3/depth/eth_btc)

Sample response:

```
{
    "eth_btc":{
        "asks":[
            [103.426,0.01],
            [103.5,15],
            [103.504,0.425],
            [103.505,0.1],
            ...
        ],
        "bids":[
            [103.2,2.48502251],
            [103.082,0.46540304],
            [102.91,0.99007913],
            [102.83,0.07832332],
            ...
        ]
    }
    ...
}
```

asks: Sell orders.
bids: Buy orders.

Method trades

This method provides the information about the last trades. Additionally it accepts an optional GET-parameter limit, which indicates how many orders should be displayed (150 by default). The maximum allowable value is 2000.

Sample request: https://api.liqui.io/api/3/trades/eth_btc (https://api.liqui.io/api/3/trades/eth_btc)

Sample response:

```
{
   "eth_btc":[
      {
         "type":"ask",
         "price":103.6,
         "amount":0.101,
         "tid":4861261,
         "timestamp":1370818007
      },
      {
         "type":"bid",
         "price":103.989,
         "amount":1.51414,
         "tid":4861254,
         "timestamp":1370817960
      },
      ...
   ]
   ...
}
```

type: ask – Sell, bid – Buy.
price: Buy price/Sell price.
amount: the amount of asset bought/sold.
tid: trade ID.
timestamp: UTC time of the trade.

Liqui trade API

This API allows to trade on the exchange and receive information about the account.

To use this API, you need to create an API key. An API key can be created in your Profile in the API Keys section. After creating an API key you'll receive a key and a secret. Note that the Secret can be received only during the first 3 hours after the creation of the Key. API key information is used for authentication.

All requests to Trade API come from the following URL: https://api.liqui.io (https://api.liqui.io). The method name is sent via the POST-parameter method. All method parameters are sent via the POST-parameters. All server responses are received in the JSON format. Each request needs an authentication. You can find out more on authentication in the relevant section of this documentation.

In the case of successful request, the response will be of the following type:

```
{
   "success":1,
   "return":{<response>}
}
```

Response in the case of error:

```
{
   "success":0,
   "error":""
}
```

In the case of error you may also receive a response not in the JSON format. It usually happens if API limits are exceeded or in the case of unknown errors.

Authentication

Authentication is made by sending the following HTTP headers:

Key — API key. API key examples: 46G9R9D6-WJ77XOIP-XH9HH5VQ-A3XN3YOZ-8T1R8I8T. API keys are created in the Profile in the API keys section.

Sign — Signature. POST-parameters (?nonce=1&param0=val0), signed with a Secret key using HMAC-SHA512. For successful authentication you need to send a POST-parameter nonce with incremental numeric value for each request.

Example of using nonce values:

1. request: nonce=1
2. request: nonce=2

3. request: nonce=10
4. request: nonce=10 — an error will be displayed, because nonce is equal to the previous request
5. request: nonce=11
6. request: nonce=9 — an error will be displayed, because nonce is smaller than the nonce value in the API key

Minimum nonce value - 1, maximum - 4294967294. To reset the nonce value you need to create a new key.

Method getInfo

Returns information about the user’s current balance, API-key privileges, the number of open orders and Server Time. To use this method you need a privilege of the key info.

Parameters: None.

Sample response:

```
{
   "success":1,
   "return":{
   "funds":{
      "eth":325,
      "btc":23.998,
      "ltc":0,
      ...
      },
   "rights":{
      "info":1,
      "trade":0,
      "withdraw":0
      },
   "transaction_count":0,
      "open_orders":1,
       "server_time":1342123547
      }
}
```

funds: Your account balance available for trading. Doesn’t include funds on your open orders.
rights: The privileges of the current API key. At this time the privilege to withdraw is not used anywhere..
transaction_count: Deprecated, is equal to 0.
open_orders: The number of your open orders.
server_time: Server time (UTC).

Method Trade

The basic method that can be used for creating orders and trading on the exchange. To use this method you need an API key privilege to trade. You can only create limit orders using this method, but you can emulate market orders using rate parameters. E.g. using rate=0.1 you can sell at the best market price. Each pair has a different limit on the minimum / maximum amounts, the minimum amount and the number of digits after the decimal point. All limitations can be obtained using the info method in PublicAPI v3.

Parameters:

Parameter	description	assumes value
pair	pair	eth_btc (example)
type	order type	buy or sell
rate	the rate at which you need to buy/sell	numerical
amount	the amount you need to buy / sell	numerical

You can get the list of pairs using the info method in PublicAPI v3.

Sample response

```
{
   "success":1,
   "return":{
      "received":0.1,
      "remains":0,
      "order_id":0,
      "funds":{
         "eth":325,
         "btc":2.498,
         "ltc":0,
         ...
      }
   }
}
```

received: The amount of currency bought/sold.
remains: The remaining amount of currency to be bought/sold (and the initial order amount).
order_id: Is equal to 0 if the request was fully “matched” by the opposite orders, otherwise the ID of the executed order will be returned.
funds: Balance after the request.

Method ActiveOrders

Returns the list of your active orders. To use this method you need a privilege of the info key. If the order disappears from the list, it was either executed or canceled.

Optional parameters:

Parameter	description	assumes value	standard value
pair	pair	eth_btc (example)	all pairs

You can get the list of pairs using the info method in PublicAPI v3.

Sample response:

```
{
   "success":1,
   "return":{
      "343152":{
         "pair":"eth_btc",
         "type":"sell",
         "amount":12.345,
         "rate":485,
         "timestamp_created":1342448420,
          "status":0
         },
      ...
 }
```

Array key: Order ID.
pair: The pair on which the order was created.
type: Order type, buy/sell.
amount: The amount of currency to be bought/sold.
rate: Sell/Buy price.
timestamp_created: The time when the order was created
status: Deprecated, is always equal to 0.

Method OrderInfo

Returns the information on particular order. To use this method you need a privilege of the info key.

Parameters:

Parameter	description	assumes value
order_id	Order ID	numerical

Sample response:

```
{
    "success":1,
    "return":{
        "343152":{
          "pair":"eth_btc",
          "type":"sell",
          "start_amount":13.345,
          "amount":12.345,
          "rate":485,
          "timestamp_created":1342448420,
          "status":0
        }
    }
}
```

Array key: Order ID.

pair: The pair on which the order was created.

type: Order type, buy/sell.

start_amount: The initial amount at the time of order creation.

amount: The remaining amount of currency to be bought/sold.

rate: Sell/Buy price.

timestamp_created: The time when the order was created.

status: 0 - active, 1 – executed order, 2 - canceled, 3 – canceled, but was partially executed.

Method CancelOrder

This method is used for order cancelation. To use this method you need a privilege of the trade key.

Parameters:

Parameter	description	assumes value
order_id	order ID	numerical

Sample response:

```
{
    "success":1,
    "return":{
        "order_id":343154,
        "funds":{
        "eth":325,
        "btc":24.998,
        "ltc":0,
        ...
    }
}
```

order_id: The ID of canceled order.

funds: Balance upon request.

Method TradeHistory

Returns trade history. To use this method you need a privilege of the info key.

Optional parameters:

Parameter	description	assumes value	standard value
from	trade ID, from which the display starts	numerical	0
count	the number of trades for display	numerical	1000
from_id	trade ID, from which the display starts	numerical	0
end_id	trade ID on which the display ends	numerical	∞

order	Sorting	ASC or DESC	DESC
since	the time to start the display	UTC time	0
end	the time to end the display	UTC time	∞
pair	pair to be displayed	eth_btc (example)	all pairs

When using parameters since or end, the order parameter automatically assumes the value ASC. When using the since parameter the maximum time that can displayed is 1 week.

Sample response::

```
{
  "success":1,
  "return":{
    "166830":{
      "pair":"eth_btc",
      "type":"sell",
      "amount":1,
      "rate":450,
      "order_id":343148,
      "is_your_order":1,
      "timestamp":1342445793
    }
  }
}
```

Array keys: Trade ID.
pair: The pair on which the trade was executed.
type: Trade type, buy/sell.
amount: The amount of currency was bought/sold.
rate: Sell/Buy price.
order_id: Order ID.
is_your_order: Is equal to 1 if order_id is your order, otherwise is equal to 0.
timestamp: Trade execution time.

WithdrawCoin Method (not active now, will be available soon)

The method is designed for cryptocurrency withdrawals. You need to create the API key that you are going to use for this method in advance. Please provide the first 8 characters of the key (e.g. HKG82W66) in your ticket to support. We'll enable the Withdraw privilege for this key. When using this method, there will be no additional confirmations of withdrawal. Please note that you are fully responsible for keeping the secret of the API key safe after we have enabled the Withdraw privilege for it.

Parameters:

Parameter	Description	Assumes value
coinName	currency	BTC, LTC (example)
amount	withdrawal amounta	numeric
address	withdrawal address	address

Sample response:

```
{
  "success":1,
  "return":{
    "tId":37832629,
    "amountSent":0.009,
    "funds":{
      "eth":325,
      "btc":24.998,
      "ltc":0,
      ...
    }
  }
}
```

tId: Transaction ID.
amountSent: The amount sent including commission.
funds: Balance after the request.

Server Time	Users Online	Volume 24-hours
April 05, 2018, 07:54	1759 of 276886	4583.56 BTC

About (/About) API (/api) Fees & Limits (/fee) Support Terms of Use (/Terms)