**Contents**

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 hour after the creation of the Key.

API key information is used for authentication.

All requests to Trade API come from the following URL: https://btc-e.com/tapi

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":"<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.

**Examples**

Most of the examples have been created by our users. We are not responsible for their performance and do not provide support for them. You can use them at your own risk.

If you want to see your example here, please create a ticket titled "API example for documentation".

Examples list:

**PHP:** <http://pastebin.com/8fbMCguM>

**PHP:** <https://github.com/marinu666/PHP-btce-api> by marinu666

**Python:** <http://pastebin.com/ec11hxcP> by miraclemax

**Python:** <https://github.com/alanmcintyre/btce-api> by alanmcintyre

**Python:** <https://github.com/t0pep0/btc-e.api.python> by t0pep0

**Python:** <http://pastebin.com/kABSEyYB> by stozher

**Java:** <http://pastebin.com/jyd9tACF> by dApZoKntut

**Java:** <https://github.com/alexandersjn/btc_e_assist_api> by alexandersjn

**C#:** <https://github.com/DmT021/BtceApi> by DmT

**C++/CLI:** <http://pastebin.com/YvxmCRL9> by PoorGirl

**VB.NET:** <http://pastebin.com/JmJZSsd7> by franky1

**Objective-C:** <https://github.com/backmeupplz/BTCEBot> by backmeupplz

**Ruby:** <https://github.com/cgore/ruby-btce> by cgore & teddythetwig

**Swift:** <https://bitbucket.org/MaximSh/yawzabot-btce-bot> by MaximSh

**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":{

"usd":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 (MSK).

**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 | btc\_usd (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":{

"usd":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 | btc\_usd (example) | all pairs |

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

**Sample response:**

{

"success":1,

"return":{

"343152":{

"pair":"btc\_usd",

"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":"btc\_usd",

"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":{

"usd":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 | UNIX time | 0 |
| end | the time to end the display | UNIX time | ∞ |
| pair | pair to be displayed | btc\_usd (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":"btc\_usd",

"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.

**Method TransHistory**

Returns the history of transactions.

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

**Optional parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | description | assumes value | standard value |
| from | transaction ID, from which the display starts | numerical | 0 |
| count | number of transaction to be displayed | numerical | 1000 |
| from\_id | transaction ID, from which the display starts | numerical | 0 |
| end\_id | transaction ID on which the display ends | numerical | ∞ |
| order | sorting | ASC or DESC | DESC |
| since | the time to start the display | UNIX time | 0 |
| end | the time to end the display | UNIX time | ∞ |

When using the parameters *since* or *end*, the *order* parameter automatically assumes the value *ASC*.

**Sample response:**

{

"success":1,

"return":{

"1081672":{

"type":1,

"amount":1.00000000,

"currency":"BTC",

"desc":"BTC Payment",

"status":2,

"timestamp":1342448420

}

}

}

**Array keys**: Transaction ID.

**type**: Transaction type. 1/2 - deposit/withdrawal, 4/5 - credit/debit.

**amount**: Transaction amount.

**currency**: Transaction currency.

**desc**: Transaction description.

**status**: Transaction status. 0 - canceled/failed, 1 - waiting for acceptance, 2 - successful, 3 – not confirmed

**timestamp**: Transaction time.