accounts@coinapult.com
coinapult.com

COINAPULT

Coinapult API Specification

Copyright 2013, 2014 Coinapult, all rights reserved v1.07.02
Author: Ira Miller, Guilherme Polo 3/31/2014

Summary

Coinapult makes all core wallet functionality, and additional merchant tools available via API. The API is RESTful and all the returned messages are in JSON.

Authentication

Most API functions require authentication. To authenticate a request, the following steps should be followed

1. JSON encode params, sorting alphabetically by key and making sure every value is a string
2. Create HMAC of JSON encoded params using your API secret and SHA512
3. Add HMAC to the request header field 'cpt-hmac'
4. Add API key to the request header field 'cpt-key'
5. Include a unique 'nonce' argument in the POST content
6. Include the current unix timestamp as 'timestamp' in the POST content

Note that the timestamp must be less than 1 hour old, and for all purposes the nonce must be unique within an API key. If either of these is not the case, or the HMAC doesn't match expectations, an HTTP 401 error will be returned.

Errors

If any errors occur when handling an API request, the response will follow this format:

```
{'error':'error message describing problem goes here'}
```

Any response that contains the 'error' field should be considered unsuccessful for the reason given. Note that HTTP errors may also be thrown depending on the issue.

Callbacks

If a callback URL is specified in the transaction, a callback will be made when the transaction enters a final state (complete, canceled). Each callback will be a JSON encoded transaction record sent via POST. Here are two examples:

```
{
        "transaction_id":"52289f703b57f529911fe8e3",
        "timestamp":1378393969,
        "quote":{
                "ask": 124.11,
                "bid": 123.72
        },
        "in":{
                "expected": 10.0,
                "currency": "BTC",
                "amount": 10.0
        },
        "state": "complete",
        "expiration": 1378394868,
        "out": {
                "expected": 1235.96,
                "currency": "USD",
                "amount": 1235.96
        },
        "type": "invoice",
        "address": "1FvDD4YqVac9Zk7FHrD5EkkJdMBJVeh4ho",
},
{
        "transaction_id": "52fd3a146f2cc034937a66ae",
        "timestamp": 1363382104,
        "in": {
                "expected": 14.15,
                "currency": “BTC”,
                "amount": 0.0
        },
        "state": "canceled",
        "out": {
                "expected": 0.0,
                "currency": "BTC",
                "amount": 0.0
        }
        "type": "payment"
}
```

If there are any errors, the callback will include an 'errors' list, describing the problems encountered, and the state should be 'canceled.' We will sign each request with your key and generate an HMAC using your secret. To authenticate the callback, follow these steps:

1. Compare API key to header field 'cpt-key'
2. Create HMAC of JSON encoded callback using your API secret and SHA512
3. Compare generated HMAC to header field 'cpt-hmac'

We will not include a nonce in the callback, but will send a timestamp.

Merchant Flow Example

Typically, a merchant wishing to accept payment will only need two functions: Create Invoice and Search. The flow would go as such:

1. User submits order on merchant site
2. Merchant makes Receive call to Coinapult and saves the returned transaction_id
3. Merchant displays returned bitcoin address, amount, and expiration deadline to the user
4. Customer pays invoice using Bitcoin network or pre-existing balance
5. Coinapult sends Merchant a callback confirming the payment

Private API Functions

All the calls to private functions need to be signed and sent as POST requests. Unless explicitly stated otherwise, the following currencies are supported for API calls:

- BTC
- USD
- EUR
- CNY

Receive - [/api/t/receive]
The default action is to specify the input amount and currency. By also using the outAmount and outCurrency fields, conversion transactions can be made.

For instance, a transaction with currency=BTC, outCurrency=USD, and outAmount=100 would result in an invoice to be paid in bitcoins, and the recipient is to receive $100. The system will create a quote at current market rate, locking in the price for 15 minutes.

The response will follow the same format as a callback. The amount to be paid is the response['in']['expected'] amount.

params:

- timestamp (UTC, unix style)
- nonce (string)
- currency (BTC only)
- outCurrency (optional)
- method (internal, bitcoin)
- amount (float) (optional)
- outAmount (float) (optional)
- callback (url)(optional)

return:

- transaction (json) – A scrubbed transaction as in the above examples

Send - [/api/t/send] - Send a specified amount to a given address
The recipient will always be paid in BTC, so 'currency' refers to which currency you will be paying for the transaction with. Similarly, amount is the amount of the currency specified you wish to send.

params:

- timestamp (UTC, unix style)
- nonce (unique within 24 hours)
- currency (currency you will pay in)
- amount (float)
- address (recipient's address)
- callback (url)(optional)

return:

- transaction (json) – A scrubbed transaction as in the above examples

Convert - [/api/t/convert] - Convert an amount between two currencies

params:

- timestamp (UTC, unix style)
- nonce (unique within 24 hours)
- inCurrency
- outCurrency
- amount (float)
- callback (url) (optional)

return:

- transaction (json) – A scrubbed transaction as in the above examples

Search - [/api/t/search] - Search transaction history by common keys

params:

- timestamp (UTC, unix style)
- nonce (unique within 24 hours)
- transaction_id (optional)
- type (invoice, payment, conversion, etc.) (optional)
- currency (optional)
- to (optional)
- from (optional)
- txhash (string) (optional)
- many (boolean) (optional)
- page (int) (optional)

return:

- callback style transaction

By default, transaction search is restricted to returning a single transaction that matches the specified criteria according to the combination of ‘transaction_id’, ‘type’, ‘currency’, ‘to’, ‘from’, and ‘txhash’ parameters. The last one refers to the txid returned by certain bitcoind calls.

To search for multiple transactions, set the ‘many’ field to ‘1’. When searching for multiple transactions, the response will include a ‘pageCount’ field determining how many other pages of results you can look for, and also a ‘results’ field with a list of the returned transactions. To specify a page, set the ‘page’ field to the desired integer number.

Get Bitcoin Address - [/api/getbitcoinaddress]

Get a new receiving Bitcoin address for your account.

params:

return:

- address – a bitcoin address for your account

Public API Functions

All the calls to public functions need to be sent as GET requests.

Get exchange rates - [/api/getRates] *deprecated*

Get recommended retail exchange rates of all supported pairs.

params:

return:

- (timestamp)
  - ask
    - (int)
    - (float)
  - bid
    - (int)
    - (float)

Ticker – [/api/ticker]

Get recommended retail exchange rates of all supported pairs at different levels.

Params:

- market: (USD/BTC, EUR/BTC, CNY/BTC) (string) (optional)

return:

If 'begin' and 'end' are not specified, then

- updatetime: timestamp (UTC, unix style)
- index: (float)
- market: (string)

For each key in small, medium, large, vip, vip+, 100, 500, 2000, 5000, 10000

  - ask: (float)
  - bid: (float)

If `begin’ or `end’ are specified, then a JSON object of n items where each item is composed by

  - updatetime: timestamp (UTC, unix style)
  - ask: (float)
  - bid: (float)