python-kucoin Documentation

Release 0.2.0

Sam McHardy

Oct 04, 2021

Contents

Last Updated 4th Oct 2021

This is an unofficial Python wrapper for the Kucoin exchanges REST and Websocket API v2. I am in no way affiliated with Kucoin, use at your own risk.

PyPi https://pypi.python.org/pypi/python-kucoin

Source code https://github.com/sammchardy/python-kucoin

Documentation https://python-kucoin.readthedocs.io/en/latest/

Blog with examples https://sammchardy.github.io

CHAPTER 1

Features

- Implementation of REST endpoints
- Simple handling of authentication
- Response exception handling
- Implement websockets (note only python3.5+)

CHAPTER 2

TODO

- L2 and L3 Local Order Books

CHAPTER 3

Quick Start

Register an account with Kucoin.

To test on the Sandbox register with Kucoin Sandbox.

Generate an API Key or Generate an API Key in Sandbox and enable it.

```
pip install python-kucoin
```

```
from kucoin.client import Client

api_key = '<api_key>'
api_secret = '<api_secret>'
api_passphrase = '<api_passphrase>'

client = Client(api_key, api_secret, api_passphrase)

# or connect to Sandbox
# client = Client(api_key, api_secret, api_passphrase, sandbox=True)

# get currencies
currencies = client.get_currencies()

# get market depth
depth = client.get_order_book('KCS-BTC')

# get symbol klines
klines = client.get_kline_data('KCS-BTC')

# get list of markets
markets = client.get_markets()

# place a market buy order
order = client.create_market_order('NEO', Client.SIDE_BUY, size=20)
```

(continues on next page)

(continued from previous page)

```
# get list of active orders
orders = client.get_active_orders('KCS-BTC')
```

CHAPTER 4

Websockets

Note only for python3.5+

```
import asyncio

from kucoin.client import Client
from kucoin.asyncio import KucoinSocketManager

api_key = '<api_key>'
api_secret = '<api_secret>'
api_passphrase = '<api_passphrase>'


async def main():
    global loop

    # callback function that receives messages from the socket
    async def handle_evt(msg):
        if msg['topic'] == '/market/ticker:ETH-USDT':
            print(f'got ETH-USDT tick:{msg["data"]}')

        elif msg['topic'] == '/market/snapshot:BTC':
            print(f'got BTC market snapshot:{msg["data"]}')

        elif msg['topic'] == '/market/snapshot:KCS-BTC':
            print(f'got KCS-BTC symbol snapshot:{msg["data"]}')

        elif msg['topic'] == '/market/ticker:all':
            print(f'got all market snapshot:{msg["data"]}')

        elif msg['topic'] == '/account/balance':
            print(f'got account balance:{msg["data"]}')

        elif msg['topic'] == '/market/level2:KCS-BTC':
            print(f'got L2 msg:{msg["data"]}')
```

(continues on next page)

(continued from previous page)

```
        elif msg['topic'] == '/market/match:BTC-USDT':
            print(f'got market match msg:{msg["data"]}')

        elif msg['topic'] == '/market/level3:BTC-USDT':
            if msg['subject'] == 'trade.l3received':
                if msg['data']['type'] == 'activated':
                    # must be logged into see these messages
                    print(f"L3 your order activated: {msg['data']}")
                else:
                    print(f"L3 order received:{msg['data']}")
            elif msg['subject'] == 'trade.l3open':
                print(f"L3 order open: {msg['data']}")
            elif msg['subject'] == 'trade.l3done':
                print(f"L3 order done: {msg['data']}")
            elif msg['subject'] == 'trade.l3match':
                print(f"L3 order matched: {msg['data']}")
            elif msg['subject'] == 'trade.l3change':
                print(f"L3 order changed: {msg['data']}")

    client = Client(api_key, api_secret, api_passphrase)

    ksm = await KucoinSocketManager.create(loop, client, handle_evt)

    # for private topics such as '/account/balance' pass private=True
    ksm_private = await KucoinSocketManager.create(loop, client, handle_evt,␣
↪private=True)

    # Note: try these one at a time, if all are on you will see a lot of output

    # ETH-USDT Market Ticker
    await ksm.subscribe('/market/ticker:ETH-USDT')
    # BTC Symbol Snapshots
    await ksm.subscribe('/market/snapshot:BTC')
    # KCS-BTC Market Snapshots
    await ksm.subscribe('/market/snapshot:KCS-BTC')
    # All tickers
    await ksm.subscribe('/market/ticker:all')
    # Level 2 Market Data
    await ksm.subscribe('/market/level2:KCS-BTC')
    # Market Execution Data
    await ksm.subscribe('/market/match:BTC-USDT')
    # Level 3 market data
    await ksm.subscribe('/market/level3:BTC-USDT')
    # Account balance - must be authenticated
    await ksm_private.subscribe('/account/balance')

    while True:
        print("sleeping to keep loop open")
        await asyncio.sleep(20, loop=loop)


if __name__ == "__main__":

    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
```

For more check out the documentation.

CHAPTER 5

Donate

If this library helped you out feel free to donate.

- ETH: 0xD7a7fDdCfA687073d7cC93E9E51829a727f9fE70
- NEO: AVJB4ZgN7VgSUtArCt94y7ZYT6d5NDfpBo
- LTC: LPC5vw9ajR1YndE1hYVeo3kJ9LdHjcRCUZ
- BTC: 1Dknp6L6oRZrHDECRedihPzx2sSfmvEBys

CHAPTER 6

Other Exchanges

If you use Binance check out my python-binance library.

If you use Binance Chain check out my python-binance-chain library.

If you use Allcoin check out my python-allcoin library.

If you use IDEX check out my python-idex library.

If you use BigONE check out my python-bigone library.

6.1 Contents

6.1.1 Getting Started

This API has been updated to work with the v2 Sandbox

Installation

`python-kucoin` is available on PYPI. Install with `pip`:

```
pip install python-kucoin
```

For previous v1 API install with

```
pip install python-kucoin==0.1.12
```

Register on Kucoin

Firstly register an account with Kucoin.

To test on the Sandbox register with Kucoin Sandbox.

Generate an API Key

To use signed account methods you are required to create an API Key and enable it.

Initialise the client

Pass your API Key, Secret and API Passphrase

```
from kucoin.client import Client
client = Client(api_key, api_secret, api_passphrase)
```

Requests Settings

python-kucoin uses the requests library.

You can set custom requests parameters for all API calls when creating the client.

```
client = Client("api-key", "api-secret", "api-passphrase", {"verify": False, "timeout
↪": 20})
```

Check out the requests documentation for all options.

Proxy Settings

You can use the Requests Settings method above

```
proxies = {
    'http': 'http://10.10.1.10:3128',
    'https': 'http://10.10.1.10:1080'
}

# in the Client instantiation
client = Client("api-key", "api-secret", {'proxies': proxies})
```

Or set an environment variable for your proxy if required to work across all requests.

An example for Linux environments from the requests Proxies documentation is as follows.

```
$ export HTTP_PROXY="http://10.10.1.10:3128"
$ export HTTPS_PROXY="http://10.10.1.10:1080"
```

For Windows environments

```
C:\>set HTTP_PROXY=http://10.10.1.10:3128
C:\>set HTTPS_PROXY=http://10.10.1.10:1080
```

API Rate Limit

Public Endpoints - 30 requests per ten seconds.

Private Endpoints - 50 requests per ten seconds.

- Websocket *

Connect - 30 times per minutes

Subscribe - 120 times per minute

Unsubscribe - 120 times per minute

6.1.2 Constants

Kucoin defines constants for Redord Types, Order Side, Order Status and Resolution. These are accessible from the Client class.

```
SIDE_BUY = 'buy'
SIDE_SELL = 'sell'

ACCOUNT_MAIN = 'main'
ACCOUNT_TRADE = 'trade'


ORDER_LIMIT = 'limit'
ORDER_MARKET = 'market'
ORDER_LIMIT_STOP = 'limit_stop'
ORDER_MARKET_STOP = 'market_stop'

STOP_LOSS = 'loss'
STOP_ENTRY = 'entry'

STP_CANCEL_NEWEST = 'CN'
STP_CANCEL_OLDEST = 'CO'
STP_DECREASE_AND_CANCEL = 'DC'
STP_CANCEL_BOTH = 'CB'

TIMEINFORCE_GOOD_TILL_CANCELLED = 'GTC'
TIMEINFORCE_GOOD_TILL_TIME = 'GTT'
TIMEINFORCE_IMMEDIATE_OR_CANCEL = 'IOC'
TIMEINFORCE_FILL_OR_KILL = 'FOK'
```

Use in your code like below.

```
from kucoin.client import Client

order_side = Client.SIDE_BUY
```

6.1.3 General Endpoints

6.1.4 Currency Endpoints

6.1.5 Account Endpoints

6.1.6 Trading Endpoints

6.1.7 Market Endpoints

6.1.8 Sub Account Endpoints

6.1.9 Websockets

Note: The websocket client is only available for python3.5+

This feature is still in development so check the documentation around message topics here https://docs.kucoin.com/#websocket-feed

For private topics such as '/account/balance' pass private=True to the KucoinSocketManager, see example below

TODO:

- Helper functions for topics
- Multiplexing
- Local Order book level 2 & 3

Sample Code

```
import asyncio

from kucoin.client import Client
from kucoin.asyncio import KucoinSocketManager

api_key = '<api_key>'
api_secret = '<api_secret>'
api_passphrase = '<api_passphrase>'


async def main():
    global loop

    # callback function that receives messages from the socket
    async def handle_evt(msg):
        if msg['topic'] == '/market/ticker:ETH-USDT':
            print(f'got ETH-USDT tick:{msg["data"]}')

        elif msg['topic'] == '/market/snapshot:BTC':
            print(f'got BTC market snapshot:{msg["data"]}')

        elif msg['topic'] == '/market/snapshot:KCS-BTC':
            print(f'got KCS-BTC symbol snapshot:{msg["data"]}')
```

(continues on next page)

(continued from previous page)

```
        elif msg['topic'] == '/market/ticker:all':
            print(f'got all market snapshot:{msg["data"]}')

        elif msg['topic'] == '/account/balance':
            print(f'got account balance:{msg["data"]}')

        elif msg['topic'] == '/market/level2:KCS-BTC':
            print(f'got L2 msg:{msg["data"]}')

        elif msg['topic'] == '/market/match:BTC-USDT':
            print(f'got market match msg:{msg["data"]}')

        elif msg['topic'] == '/market/level3:BTC-USDT':
            if msg['subject'] == 'trade.l3received':
                if msg['data']['type'] == 'activated':
                    # must be logged into see these messages
                    print(f"L3 your order activated: {msg['data']}")
                else:
                    print(f"L3 order received:{msg['data']}")
            elif msg['subject'] == 'trade.l3open':
                print(f"L3 order open: {msg['data']}")
            elif msg['subject'] == 'trade.l3done':
                print(f"L3 order done: {msg['data']}")
            elif msg['subject'] == 'trade.l3match':
                print(f"L3 order matched: {msg['data']}")
            elif msg['subject'] == 'trade.l3change':
                print(f"L3 order changed: {msg['data']}")

    client = Client(api_key, api_secret, api_passphrase)

    ksm = await KucoinSocketManager.create(loop, client, handle_evt)

    # for private topics such as '/account/balance' pass private=True
    ksm_private = await KucoinSocketManager.create(loop, client, handle_evt,␣
↪private=True)

    # Note: try these one at a time, if all are on you will see a lot of output

    # ETH-USDT Market Ticker
    await ksm.subscribe('/market/ticker:ETH-USDT')
    # BTC Symbol Snapshots
    await ksm.subscribe('/market/snapshot:BTC')
    # KCS-BTC Market Snapshots
    await ksm.subscribe('/market/snapshot:KCS-BTC')
    # All tickers
    await ksm.subscribe('/market/ticker:all')
    # Level 2 Market Data
    await ksm.subscribe('/market/level2:KCS-BTC')
    # Market Execution Data
    await ksm.subscribe('/market/match:BTC-USDT')
    # Level 3 market data
    await ksm.subscribe('/market/level3:BTC-USDT')
    # Account balance - must be authenticated
    await ksm_private.subscribe('/account/balance')

    while True:
```

(continues on next page)

(continued from previous page)

```
        print("sleeping to keep loop open")
        await asyncio.sleep(20, loop=loop)


if __name__ == "__main__":

    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
```

Unsubscribe from channel

Send the same topic name to the unsubscribe function

```
await ksm.unsubscribe('/market/ticker:ETH-USDT')
```

6.1.10 Exceptions

KucoinResponseException

Raised if a non JSON response is returned

KucoinAPIException

On an API call error a kucoin.exceptions.KucoinAPIException will be raised.

The exception provides access to the

- *status_code* - response status code
- *response* - response object
- *message* - Kucoin error message=
- *request* - request object if available

```
try:
    client.get_coin_list()
except KucoinAPIException as e:
    print(e.status_code)
    print(e.message)
```

MarketOrderException

Raised if market order params are incorrect

LimitOrderException

Raised if limit order params are incorrect

6.1.11 Changelog

v2.1.3 - 2021-10-04

Added

- chain to get_deposit_address
- get_accounts to include currency and account_type params
- get_account_activity to use new params
- get_status to retrieve service status
- get and cancel order by orderOid
- trade_type to market and limit orders
- v2 and v3 API version options

Deprecated

- get_account_holds

v2.1.2 - 2019-04-17

Added

- exception if using a private websocket topic but not connected with private

Removed

- removed py3.6 type annotations for py3.5 support

v2.1.1 - 2019-04-17

Added

- websocket support for private messages
- *get_historical_orders* function to get V1 historical orders

Fixed

- fixed *get_ticker* to work for all tickers
- websocket reconnect ability

v2.1.0 - 2019-02-25

Added

- websocket support
- *get_fiat_prices* function to get fiat price for currency
- *get_markets* function to get supported market list
- iceberg order support
- util functions to generate uuid and convert dict to compact json string

Updated

- *get_ticker* to have optional symbol param

Fixed

- market and limit order create functions
- *get_kline_data* function
- *get_account_holds* function endpoint
- LimitOrderException message

v2.0.2 - 2019-02-23

Fixed

- signature generation for get requests

v2.0.1 - 2019-01-23

Fixed

- added auth for get_fills()

v2.0.0 - 2019-01-22

Added

- support for REST endpoint of v2 API

v0.1.12 - 2018-04-27

Added

- timestamp in milliseconds to *get_historical_klines_tv* function

Fixed

- make *coin* parameter required in *get_coin_info* function

v0.1.11 - 2018-03-01

Added

- option for passing requests module parameters on Client initialisation

Restored

- old *get_all_balances* non-paged functionality

v0.1.10 - 2018-02-10

Fixed

- remove slash in path in *get_order_details* function

v0.1.9 - 2018-02-09

Updated

- path for *get_all_balances* to match update in Kucoin docs, now supports pagination

v0.1.8 - 2018-01-20

Added

- better exception error messages

Fixed

- *cancel_order* format to make *order_type* required

v0.1.7 - 2018-01-17

Fixed

- *cancel_order* format to send symbol in payload, remove URL params
- *cancel_all_orders* format to send symbol in payload, remove URL params

v0.1.6 - 2018-01-15

Added

- constants for transfer types, pending, finished and cancelled
- documentation for *group* param on *get_order_book*, *get_buy_orders* and *get_sell_orders*
- add *get_trading_markets* endpoint
- add *market* param to *get_trading_symbols* and *get_trending_coins*
- add *get_coin_info* function with optional *coin* param

Fixed

- set coin param to optional for *get_reward_info*, *get_reward_summary* and *extract_invite_bonus*
- actually use the *kv_format* param on *get_active_orders*
- *cancel_order* format to send symbol in URL
- *cancel_all_orders* format to send symbol in URL
- *order_details* removed symbol from URL
- *get_tick* symbol is now optional
- fix *get_coin_list* URL

v0.1.5 - 2018-01-14

Fixed

- remove debug output

v0.1.4 - 2018-01-14

Added

- add function *get_historical_klines_tv* to get klines in OHLCV format

Fixed

- handle success: false type errors properly to raise exception
- fix passed param name on *get_kline_data*

v0.1.3 - 2018-01-12

Added

- add function *get_total_balance* to get balance in Fiat
- added pagination params to *get_all_balances*

v0.1.2 - 2018-01-07

Added

- api key endpoints
- set default currency function
- extract invite bonus function

v0.1.1 - 2018-01-02

Added

- cancel all orders function
- get order details function
- get dealt orders function

Updated

- old get_deal_orders function to get_symbol_dealt_orders

v0.1.0 - 2017-11-12

Added

- Kucoin client interface
- Coverage for all main endpoints
- Constants for transfer type and status, order side and kline resolution
- Full documentation

6.1.12 Kucoin API

client module

exceptions module

exception kucoin.exceptions.**KucoinAPIException**(*response*)
Bases: exceptions.Exception

Exception class to handle general API Exceptions

code values

message format

__init__(*response*)
x.__init__(...) initializes x; see help(type(x)) for signature

exception kucoin.exceptions.**KucoinRequestException**(*message*)
Bases: exceptions.Exception

__init__(*message*)
x.__init__(...) initializes x; see help(type(x)) for signature

exception kucoin.exceptions.**MarketOrderException**(*message*)
Bases: exceptions.Exception

__init__(*message*)
x.__init__(...) initializes x; see help(type(x)) for signature

exception kucoin.exceptions.**LimitOrderException**(*message*)
Bases: exceptions.Exception

__init__(*message*)
x.__init__(...) initializes x; see help(type(x)) for signature

6.2 Index

- genindex

Python Module Index

k

Index

Symbols

K

L

M