

---

# **python-cryptowatch Documentation**

***Release 0.0.1***

**uoshvis**

**Oct 10, 2021**



---

## Contents

---

<b>1 Quick Start</b>	<b>3</b>
1.1 Contents . . . . .	4
1.2 Indices and tables . . . . .	15
<b>Python Module Index</b>	<b>17</b>
<b>Index</b>	<b>19</b>



This is an unofficial Python wrapper for the [Cryptowatch public Data API](#). Cryptowatch is a cryptocurrency charting and trading platform owned by [Kraken exchange](#).

**Source code** <https://github.com/uoshvis/python-cryptowatch>

**Documentation** <https://python-cryptowatch.readthedocs.io/en/latest/>



# CHAPTER 1

---

## Quick Start

---

```
from cryptowatch.api_client import Client
client = Client()

# get assets
assets = client.get_assets()

# get markets which have btc as base or quote
assets_btc = client.get_assets('btc')

"""
Returns a market's OHLC candlestick data.
This represents a 1-hour candle starting at 1594087200 (Tuesday,
7 July 2020 02:00:00 GMT) and ending at 1602179348 (Thursday,
8 October 2020 17:49:08 GMT).
"""

data = {
    'exchange': 'gdax',
    'pair': 'btcusd',
    'route': 'ohlcv',
    'params': {
        'before': 1602179348,
        'after': 1594087200,
        'periods': '3600'
    }
}

market = client.get_markets(data=data)
```

For more check out the [documentation](#).

## 1.1 Contents

### 1.1.1 Getting Started

#### Initialise the client

```
from cryptowatch.api_client import Client
client = Client()
```

### 1.1.2 Available Endpoints

#### Get assets

```
client.get_assets()
```

#### Get pairs

```
client.get_pairs()
```

#### Get exchanges

```
client.get_exchanges()
```

#### Get markets

```
client.get_markets()
```

#### Get aggregates

```
client.get_aggregates()
```

### 1.1.3 Exceptions

#### CryptowatchAPIException

On an API call error a `binance.exceptions.BinanceAPIException` will be raised.

The exception provides access to the

- *status\_code* - response status code
- *reason* - response reason
- *response* - response object



## CryptowatchResponseException

Raised if a non JSON response is returned

## 1.1.4 Cryptowatch API

### api\_client module

Module related to the client interface to cryptowat.ch API.

**class** cryptowatch.api\_client.**Client**

Bases: object

The public client to the cryptowat.ch api.

**API\_URL** = 'https://api.cryptowat.ch'

**ROUTESAggregate** = ['prices', 'summaries']

**ROUTESMarket** = ['price', 'summary', 'orderbook', 'trades', 'ohlcv']

**ROUTESParams** = ['trades', 'ohlcv']

**get\_aggregates** (\*args)

Retrieves the prices and summaries of all markets on the site in a single request.

#### Prices

To get the current price for all supported markets use:

```
get_aggregates('prices')
```

#### Returns API response

```
{
  "result": {
    {
      "bitfinex:bfxbtc": 0.00067133,
      "bitfinex:bfxusd": 0.52929,
      "bitfinex:btcusd": 776.73,
      ...
    }
  }
}
```

#### Summaries

To get the market summary for all supported markets use:

```
get_aggregates('summaries')
```

#### Returns API response

```
{
  "result": {
    {
      "bitfinex:bfxbtc": {
```

(continues on next page)

(continued from previous page)

```
        "price": {
            "last": 0.00067133,
            "high": 0.0006886,
            "low": 0.00066753,
            "change": {
                "percentage": -0.02351996,
                "absolute": -1.6169972e-05
            }
        },
        "volume": 84041.625
    },
    "bitfinex:bfusd": {
        ...
    },
    "bitfinex:btcusd": {
        ...
    },
    ...
}
```

**get\_assets** (*asset=None*)

An asset can be a crypto or fiat currency.

**Index**

```
get_assets()
```

**Returns** API response

```
{
  "result": [
    {
      "symbol": "aud",
      "name": "Australian Dollar",
      "fiat": true,
      "route": "https://api.cryptowat.ch/assets/aud"
    },
    {
      "symbol": "etc",
      "name": "Ethereum Classic",
      "fiat": false,
      "route": "https://api.cryptowat.ch/assets/etc"
    },
    ...
  ]
}
```

**Asset**

Returns a single asset. Lists all markets which have this asset as a base or quote.

```
get_assets('btc')
```

**Returns** API response

```
{
  "result": {
    "symbol": "btc",
    "name": "Bitcoin",
    "fiat": false,
    "markets": {
      "base": [
        {
          "exchange": "bitfinex",
          "pair": "btcusd",
          "active": true,
          "route": "https://api.cryptowat.ch/markets/bitfinex/btcusd"
        },
        {
          "exchange": "gdax",
          "pair": "btcusd",
          "route": "https://api.cryptowat.ch/markets/gdax/btcusd"
        },
        ...
      ],
      "quote": [
        {
          "exchange": "bitfinex",
          "pair": "ltcbtc",
          "active": true,
          "route": "https://api.cryptowat.ch/markets/bitfinex/ltcbtc"
        },
        {
          "exchange": "bitfinex",
          "pair": "ethbtc",
          "active": true,
          "route": "https://api.cryptowat.ch/markets/bitfinex/ethbtc"
        },
        ...
      ]
    }
  }
}
```

**get\_exchanges** (*exchange=None*)

Exchanges are where all the action happens!

### Index

Returns a list of all supported exchanges.

```
get_exchanges()
```

**Returns** API response

```
{
  "result": [
    {
      "symbol": "bitfinex",
      "name": "Bitfinex",
      "active": true,
      "route": "https://api.cryptowat.ch/exchanges/bitfinex"
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```
    },
    {
        "symbol": "gdax",
        "name": "GDAX",
        "active": true,
        "route": "https://api.cryptowat.ch/exchanges/gdax"
    },
    ...
]
}
```

### Exchange

Returns a single exchange, with associated routes.

```
get_exchanges('kraken')
```

**Returns** API response

```
{
  "result": {
    "id": "kraken",
    "name": "Kraken",
    "active": true,
    "routes": {
      "markets": "https://api.cryptowat.ch/markets/kraken"
    }
  }
}
```

**get\_markets** (*path=None*, *\*\*kwargs*)

A market is a pair listed on an exchange. For example, pair btceur on exchange kraken is a market.

### Index

Returns a list of all supported markets.

```
get_markets()
```

**Returns** API response

```
{
  "result": [
    {
      "exchange": "bitfinex",
      "pair": "btcusd",
      "active": true,
      "route": "https://api.cryptowat.ch/markets/bitfinex/btcusd"
    },
    {
      "exchange": "bitfinex",
      "pair": "ltcusd",
      "active": true,
      "route": "https://api.cryptowat.ch/markets/bitfinex/ltcusd"
    },
  ],
}
```

(continues on next page)

(continued from previous page)

```
{
  "exchange": "bitfinex",
  "pair": "ltcbtc"
  "active": true,
  "route": "https://api.cryptowat.ch/markets/bitfinex/ltcbtc"
},
...
]
```

To get the supported markets for only a specific exchange:

```
get_markets('kraken')
```

### Market

Returns a single market, with associated routes.

#### Parameters

- **exchange** (*str*) – required
- **pair** (*str*) – required

```
data = {
  'exchange': 'gdax',
  'pair': 'btcusd'
}
```

```
get_markets(data=data)
```

#### Returns API response

```
{
  "result": {
    "exchange": "gdax",
    "pair": "btcusd",
    "active": true,
    "routes": {
      "price": "https://api.cryptowat.ch/markets/gdax/btcusd/price",
      "summary": "https://api.cryptowat.ch/markets/gdax/btcusd/summary",
      "orderbook": "https://api.cryptowat.ch/markets/gdax/btcusd/orderbook",
      "trades": "https://api.cryptowat.ch/markets/gdax/btcusd/trades",
      "ohlcv": "https://api.cryptowat.ch/markets/gdax/btcusd/ohlcv"
    }
  }
}
```

### Price

Returns a market's last price.

#### Parameters

- **exchange** (*str*) – required
- **pair** (*str*) – required
- **route** (*str*) – 'price'

```
data = {
    'exchange': 'gdax',
    'pair': 'btcusd',
    'route': 'price'
}
```

```
get_markets(data=data)
```

**Returns** API response

```
{
  "result": {
    "price": 780.63
  }
}
```

### Summary

Returns a market's last price as well as other stats based on a 24-hour sliding window.

#### Parameters

- **exchange** (*str*) – required
- **pair** (*str*) – required
- **route** (*str*) – ‘summary’

```
data = {
    'exchange': 'gdax',
    'pair': 'btcusd',
    'route': 'summary'
}
```

```
get_markets(data=data)
```

**Returns** API response

```
{
  "result": {
    "price": {
      "last": 780.31,
      "high": 790.34,
      "low": 772.76,
      "change": {
        "percentage": 0.0014373838,
        "absolute": 1.12
      }
    },
    "volume": 5345.0415
  }
}
```

### Trades

Returns a market's most recent trades, incrementing chronologically.

#### Parameters

- **exchange** (*str*) – required
- **pair** (*str*) – required
- **route** (*str*) – ‘trades’
- **params** (*dict*) – supported params
- **params.limit** (*int* (defaults to 50)) – limit amount of trades returned
- **params.since** (*UNIX timestamp*) – only return trades at or after this time.

```
data = {
    'exchange': 'gdax',
    'pair': 'btcusd',
    'route': 'trades',
    'params': {'limit': 10, 'since': 1481663244 }
}
```

```
get_markets(data=data)
```

### Returns API response

```
{
  "result": [
    [
      0,
      1481676478,
      734.39,
      0.1249
    ],
    [
      0,
      1481676537,
      734.394,
      0.0744
    ],
    [
      0,
      1481676581,
      734.396,
      0.1
    ],
    [
      0,
      1481676602,
      733.45,
      0.061
    ],
    ...
  ]
}
```

Trades are lists of numbers in this order:

```
[ ID, Timestamp, Price, Amount ]
```

### Order Book

To get market's order book use 'orderbook' route.

#### Parameters

- **exchange** (*str*) – required
- **pair** (*str*) – required
- **route** (*str*) – 'orderbook'

```
data = {
    'exchange': 'gdax',
    'pair': 'btcusd',
    'route': 'orderbook'
}
```

#### Returns API response

```
{
  "result": {
    "asks": [
      [
        733.73,
        2.251
      ],
      [
        733.731,
        7.829
      ],
      [
        733.899,
        1.417
      ],
      ...
    ],
    "bids": [
      [
        733.62,
        0.273
      ],
      ...
    ]
  }
}
```

Orders are lists of numbers in this order:

```
[ Price, Amount ]
```

#### OHLC

Returns a market's OHLC candlestick data. Returns data as lists of lists of numbers for each time period integer.

#### Parameters

- **exchange** (*str*) – required
- **pair** (*str*) – required
- **route** (*str*) – 'ohlc'



- **params** (*dict*) – supported params
- **params.before** (*int (defaults to 50)*) – Only return candles opening before this time
- **params.after** (*UNIX timestamp*) – Only return candles opening after this time
- **params.periods** (*Comma-separated integers 60,180,108000*) – return these time periods

```
data = {
    'exchange': 'gdax',
    'pair': 'btcusd',
    'route': 'ohlcv'
    'params': {
        'before': 1481663244,
        'after': 1481663244,
        'periods': '60,180'
    }
}
```

```
get_markets(data=data)
```

Return values are in list:

```
[ CloseTime, OpenPrice, HighPrice, LowPrice, ClosePrice, Volume, ↵
↵QuoteVolume]
```

**get\_pairs** (*pair=None*)

A pair of assets. Each pair has a base and a quote. For example, btceur has base btc and quote eur.

### Index

All pairs (in no particular order).

```
get_pairs()
```

### Returns API response

```
{
  "result": [
    {
      "symbol": "xmrusd",
      "id": 82,
      "base": {
        "symbol": "xmr",
        "name": "Monero",
        "fiat": false,
        "route": "https://api.cryptowat.ch/assets/xmr"
      },
      "quote": {
        "symbol": "usd",
        "name": "United States dollar",
        "fiat": true,
        "route": "https://api.cryptowat.ch/assets/usd"
      },
      "route": "https://api.cryptowat.ch/pairs/xmrusd"
    },
  ],
}
```

(continues on next page)

(continued from previous page)

```
{
  "symbol": "ltcUSD",
  "id": 189,
  "base": {
    "symbol": "ltc",
    "name": "Litecoin",
    "fiat": false,
    "route": "https://api.cryptowat.ch/assets/ltc"
  },
  "quote": {
    "symbol": "USD",
    "name": "United States dollar",
    "fiat": true,
    "route": "https://api.cryptowat.ch/assets/USD"
  },
  "route": "https://api.cryptowat.ch/pairs/ltcUSD"
},
...
]
```

### Pair

Returns a single pair. Lists all markets for this pair.

```
get_pairs('ethbtc')
```

### Returns API response

```
{
  "result": {
    "symbol": "ethbtc",
    "id": 23,
    "base": {
      "symbol": "eth",
      "name": "Ethereum",
      "isFiat": false,
      "route": "https://api.cryptowat.ch/assets/eth"
    },
    "quote": {
      "symbol": "btc",
      "name": "Bitcoin",
      "isFiat": false,
      "route": "https://api.cryptowat.ch/assets/btc"
    },
    "route": "https://api.cryptowat.ch/pairs/ethbtc",
    "markets": [
      {
        "exchange": "bitfinex",
        "pair": "ethbtc",
        "active": true,
        "route": "https://api.cryptowat.ch/markets/bitfinex/ethbtc"
      },
      {
        "exchange": "gdax",
        "pair": "ethbtc",

```

(continues on next page)

(continued from previous page)

```
        "active": true,  
        "route": "https://api.cryptowat.ch/markets/gdax/ethbtc"  
    },  
    ...  
]  
}  
}
```

## 1.2 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)



### C

`cryptowatch.api_client`, 5



## A

API\_URL (*cryptowatch.api\_client.Client* attribute), 5

## C

Client (*class in cryptowatch.api\_client*), 5

cryptowatch.api\_client (*module*), 5

## G

get\_aggregates () (*cryptowatch.api\_client.Client* method), 5

get\_assets () (*cryptowatch.api\_client.Client* method), 6

get\_exchanges () (*cryptowatch.api\_client.Client* method), 7

get\_markets () (*cryptowatch.api\_client.Client* method), 8

get\_pairs () (*cryptowatch.api\_client.Client* method), 13

## R

ROUTES\_AGGREGATE (*cryptowatch.api\_client.Client* attribute), 5

ROUTES\_MARKET (*cryptowatch.api\_client.Client* attribute), 5

ROUTES\_PARAMS (*cryptowatch.api\_client.Client* attribute), 5