CryptoCompare API Quick Start Guide

Alex Galea Follow

Aug 28, 2017 · 2 min read

This is the best way to get intraday trading data for cryptocurrencies. I'll run run through the most useful API functions to get current and historical intraday prices (OHLCV) on the hourly and minute time frames!

> *Last updated: August 2017*

Source Code

Check out the full code in this IPython notebook. I use these packages —

```
import requests
import datetime
import pandas as pd
import matplotlib.pyplot as plt
```

Live Coin Prices

```
def price(symbol, comparison_symbols=['USD'], exchange=''):
    url = 'https://min-api.cryptocompare.com/data/price?fsym={}&tsyms={}'\
            .format(symbol.upper(), ','.join(comparison_symbols).upper())
    if exchange:
        url += '&e={}'.format(exchange)
    page = requests.get(url)
    data = page.json()
    return data
```

```
price('LTC', exchange='Coinbase')
```

```
{'USD': 59.57}
```

```
price('NEO', ['BTC', 'ETH', 'USD'])
```

```
{'BTC': 0.00894, 'ETH': 0.1127, 'USD': 37.3}
```

Daily Historical Price (OHLCV)

```
def daily_price_historical(symbol, comparison_symbol, all_data=True, limi
    url = 'https://min-api.cryptocompare.com/data/histoday?fsym={}&tsym={
            .format(symbol.upper(), comparison_symbol.upper(), limit, agg
    if exchange:
        url += '&e={}'.format(exchange)
    if all_data:
        url += '&allData=true'
    page = requests.get(url)
    data = page.json()['Data']
    df = pd.DataFrame(data)
    df['timestamp'] = [datetime.datetime.fromtimestamp(d) for d in df.tim
    return df
```

Using the default arguments, this function will return the entirety of the available price history. To specify a row limit, pass `all_data=False` and use the `limit` parameter.

```
df = daily_price_historical('BTC', 'USD')
print('Max length = %s' % len(df))
print('Max time = %s' % (df.timestamp.max() - df.timestamp.min()))
df.head()
```

```
Max length = 2600
Max time = 2599 days 00:00:00
```

	close	high	low	open	time	volumefrom	volumeto	timestamp
0	0.04951	0.04951	0.04951	0.04951	1279324800	20.00	0.9902	2010-07-16 17:00:00
1	0.08584	0.08585	0.05941	0.04951	1279411200	75.01	5.0900	2010-07-17 17:00:00
2	0.08080	0.09307	0.07723	0.08584	1279497600	574.00	49.6600	2010-07-18 17:00:00
3	0.07474	0.08181	0.07426	0.08080	1279584000	262.00	20.5900	2010-07-19 17:00:00
4	0.07921	0.07921	0.06634	0.07474	1279670400	575.00	42.2600	2010-07-20 17:00:00

```
plt.plot(df.timestamp, df.close)
plt.xticks(rotation=45)
plt.show()
```

Hourly Historical Price (OHLCV)

```python
def hourly_price_historical(symbol, comparison_symbol, limit, aggregate,
    url = 'https://min-api.cryptocompare.com/data/histohour?fsym={}&tsym=
            .format(symbol.upper(), comparison_symbol.upper(), limit, agg
    if exchange:
        url += '&e={}'.format(exchange)
    page = requests.get(url)
    data = page.json()['Data']
    df = pd.DataFrame(data)
    df['timestamp'] = [datetime.datetime.fromtimestamp(d) for d in df.tim
    return df
```

```python
time_delta = 1 # Bar width in hours
df = hourly_price_historical('BTC', 'USD', 9999, time_delta)
print('Max length = %s' % len(df))
print('Max time = %s' % (df.timestamp.max() - df.timestamp.min()))

plt.plot(df.timestamp, df.close)
plt.xticks(rotation=45)
plt.show()
```

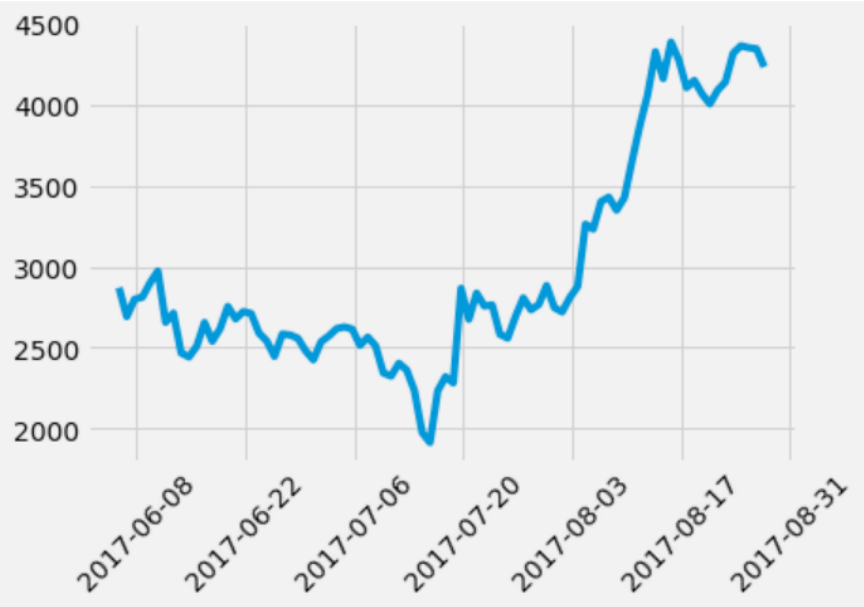
```
Max length = 2001
Max time = 83 days 08:00:00
```

Using a bar width of 1 hour

```
time_delta = 24 # Bar width in hours
df = hourly_price_historical('BTC', 'USD', 9999, time_delta)
print('Max length = %s' % len(df))
print('Max time = %s' % (df.timestamp.max() - df.timestamp.min()))

plt.plot(df.timestamp, df.close)
plt.xticks(rotation=45)
plt.show()
```

```
Max length = 84
Max time = 83 days 00:00:00
```



Using a bin width of 24 hours

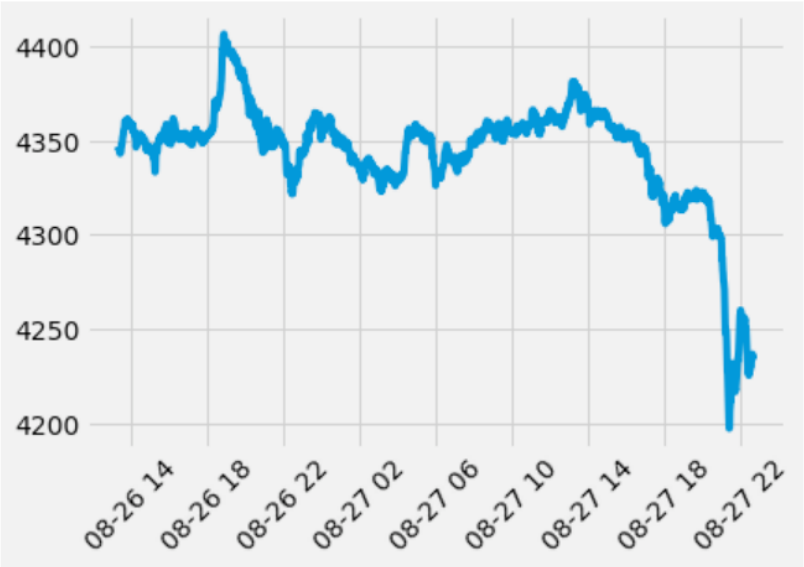
Historical Price by Minute (OHLCV)

```
def minute_price_historical(symbol, comparison_symbol, limit, aggregate,
    url = 'https://min-api.cryptocompare.com/data/histominute?fsym={}&tsy
            .format(symbol.upper(), comparison_symbol.upper(), limit, agg
    if exchange:
        url += '&e={}'.format(exchange)
    page = requests.get(url)
    data = page.json()['Data']
    df = pd.DataFrame(data)
    df['timestamp'] = [datetime.datetime.fromtimestamp(d) for d in df.tim
    return df
```

```
time_delta = 1 # Bar width in minutes
df = minute_price_historical('BTC', 'USD', 9999, time_delta)
print('Max length = %s' % len(df))
print('Max time = %s' % (df.timestamp.max() - df.timestamp.min()))

plt.plot(df.timestamp, df.close)
plt.xticks(rotation=45)
plt.show()
```

```
Max length = 2001
Max time = 1 days 09:20:00
```



Coin List

Lists out each coin (as of 2017–08 there are > 1400) and gives various details such as metrics related to the mining protocol.

```
def coin_list():
    url = 'https://www.cryptocompare.com/api/data/coinlist/'
    page = requests.get(url)
    data = page.json()['Data']
    return data
```

```
data = coin_list()
RenderJSON(data)
```

```
"RPC": ⊕{...},
```

```
"CAT": ⊕{...},
"NEO": ⊖{
    "FullName": "NEO (NEO)",
    "PreMinedValue": "N/A",
    "FullyPremined": "1",
    "TotalCoinsFreeFloat": "N/A",
    "ProofType": "N/A",
    "TotalCoinsMined": "100000000",
    "SortOrder": "718",
    "CoinName": "NEO",
    "Url": "/coins/neo/overview",
    "Id": "27368",
    "Algorithm": "N/A",
    "ImageUrl": "/media/1383858/neo.jpg",
    "Name": "NEO",
    "TotalCoinSupply": "100000000"
},
"TRICK": ⊕{...},
"COIN": ⊕{...},
"DFT": ⊕{...},
```


It's important to extract the `Id` for each coin to use in the following API calls.

```
symbol_id_dict = {symb: int(d['Id']) for symb, d in
data.items()}
```

Live Coin Information

```
def coin_snapshot_full_by_id(symbol, symbol_id_dict={}):
    if not symbol_id_dict:
        symbol_id_dict = {
            'BTC': 1182,
            'ETH': 7605,
            'LTC': 3808
        }
    symbol_id = symbol_id_dict[symbol.upper()]
    url = 'https://www.cryptocompare.com/api/data/coinsnapshotfullbyid/?i
```

```
            .format(symbol_id)
    page = requests.get(url)
    data = page.json()['Data']
```

```
data = coin_snapshot_full_by_id('ETH', symbol_id_dict)
RenderJSON(data)
```


```
{
    "ICO": {...},
    "General": {...},
    "Subs": [ ... ],
    "StreamerDataRaw": [ ... ],
    "SEO": {
        "BaseImageUrl": "https://www.cryptocompare.com",
        "PageTitle": "Ethereum (ETH) - Live Ether price and market cap",
        "OgImageHeight": "400",
        "BaseUrl": "https://www.cryptocompare.com",
        "OgImageWidth": "400",
        "OgImageUrl": "/media/20646/eth.png",
        "PageDescription": "Live Ether price from all markets and ETH coi
n market Capitalization. Stay up to date with the latest Ether price move
ments and forum discussion. Check out our snapshot charts and see when th
ere is an opportunity to buy or sell."
    }
}
```


Live Social Status

```
def live_social_status(symbol, symbol_id_dict={}):
    if not symbol_id_dict:
        symbol_id_dict = {
            'BTC': 1182,
            'ETH': 7605,
            'LTC': 3808
        }
    symbol_id = symbol_id_dict[symbol.upper()]
    url = 'https://www.cryptocompare.com/api/data/socialstats/?id={}'\
            .format(symbol_id)
    page = requests.get(url)
    data = page.json()['Data']
```

```
data = live_social_status('BTC', symbol_id_dict)
RenderJSON(data)
```


```
{
    "Facebook": {
        "link": "https://www.facebook.com/bitcoins/",
        "is_closed": "false",
        "talking_about": 154,
        "likes": 33707,
        "name": "Bitcoin P2P Cryptocurrency",
        "Points": 35247
    },
    "Reddit": {
        "comments_per_day": 3239.6,
        "link": "https://www.reddit.com/r/bitcoin/",
        "Points": 317114,
        "name": "Bitcoin",
        "comments_per_hour": "134.98",
        "posts_per_day": "373.91",
        "subscribers": 300261,
        "posts_per_hour": "15.58",
        "active_users": 3458,
        "community_creation": "1284042626"
    },
    "Twitter": {...},
    "CodeRepository": {...},
    "General": {...},
    "CryptoCompare": {...}
}
```


Conclusion

It’s pretty amazing that all this data is available through an API. Wonderful work by the people at https://www.cryptocompare.com/

Thanks for reading! If anything’s changed since I last updated this, please send me a message on twitter

@agalea91

Python Cryptocurrency API