This notebook demonstrates basic functionality offered by the Coin Metrics Python API Client and Network Data Pro.

Coin Metrics offers a vast assortment of data for hundreds of cryptoassets. The Python API Client allows for easy access to this data using Python without needing to create your own wrappers using `requests` and other such libraries.

Resources

To understand the data that Coin Metrics offers, feel free to peruse the resources below.

- The Coin Metrics API v4 website contains the full set of endpoints and data offered by Coin Metrics.
- The Coin Metrics Knowledge Base gives detailed, conceptual explanations of the data that Coin Metrics offers.
- The API Spec contains a full list of functions.

Setup

In [1]:
```
from os import environ
import sys
import pandas as pd
import numpy as np
import seaborn as sns
import logging
from datetime import date, datetime, timedelta
from coinmetrics.api_client import CoinMetricsClient
import json
import logging
from pytz import timezone as timezone_conv
from datetime import timezone as timezone_info
import matplotlib.ticker as mticker
from matplotlib.dates import DateFormatter
from matplotlib.ticker import FuncFormatter
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
%matplotlib inline
```

In [2]:
```
logging.basicConfig(
    format='%(asctime)s %(levelname)-8s %(message)s',
    level=logging.INFO,
    datefmt='%Y-%m-%d %H:%M:%S'
)
```

In [3]:
```
# We recommend privately storing your API key in your local environment.
try:
    api_key = environ["CM_API_KEY"]
    logging.info("Using API key found in environment")
except KeyError:
    api_key = ""
    logging.info("API key not found. Using community client")


client = CoinMetricsClient(api_key)
```

```
2023-10-31 15:32:59 INFO     Using API key found in environment
```

Market Cap Based on Verified On-Chain Supply

In order to trustlessly verify market capitalization, Coin Metrics directly indexes the blockchain to independently validate the amount of circulating supply.

The **CapMrktCurUSD** metric offers the most reliable measure of asset supply, with the trade-off of slightly limited asset coverage.

In [4]:
```
catalog_cur = client.catalog_asset_metrics(metrics='CapMrktCurUSD').to_dataframe()
```

In [5]:
```
catalog_cur
```

Out[5]:

	metric	full_name	description	product	category	subcategory	unit	data_type	type	display_name	frequency	asset
0	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	1inch
1	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	aave
2	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	ada
3	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	algo
4	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	alpha
...	...	...	...	...	...	...	...	...	...	...	...	...
96	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	xtz
97	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	xvg
98	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	yfi
99	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	zec
100	CapMrktCurUSD	Capitalization, market, current supply, USD	The sum USD value of the current supply. Also ...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap (USD)	1d	zrx

101 rows × 12 columns

In [6]:
```
cur_assets = catalog_cur['asset'].to_list()
```

In [7]:
```
capmrktcur = client.get_asset_metrics(
    assets=cur_assets,
    metrics='CapMrktCurUSD',
    start_time='2023-01-01'
).to_dataframe()
```

In [8]:
```
capmrktcur
```

Out[8]:

	asset	time	CapMrktCurUSD
0	1inch	2023-01-01 00:00:00+00:00	578851371.621414
1	1inch	2023-01-02 00:00:00+00:00	593808264.113895
2	1inch	2023-01-03 00:00:00+00:00	578865051.831563
3	1inch	2023-01-04 00:00:00+00:00	593238406.963806
4	1inch	2023-01-05 00:00:00+00:00	582087444.014613
...	...	...	...
26763	zrx	2023-10-26 00:00:00+00:00	236431833.032663
26764	zrx	2023-10-27 00:00:00+00:00	233984862.653407
26765	zrx	2023-10-28 00:00:00+00:00	237842249.763875
26766	zrx	2023-10-29 00:00:00+00:00	259505519.640012
26767	zrx	2023-10-30 00:00:00+00:00	265896073.884377

26768 rows × 3 columns

In [9]:
```
capmrktcur_pivot = capmrktcur.pivot(index='time',columns='asset',values='CapMrktCurUSD')
```

In [10]:
```
capmrktcur_pivot['Total Cap'] = capmrktcur_pivot.sum(axis=1)
```

In [11]:
```
capmrktcur_pivot
```

Out[11]:

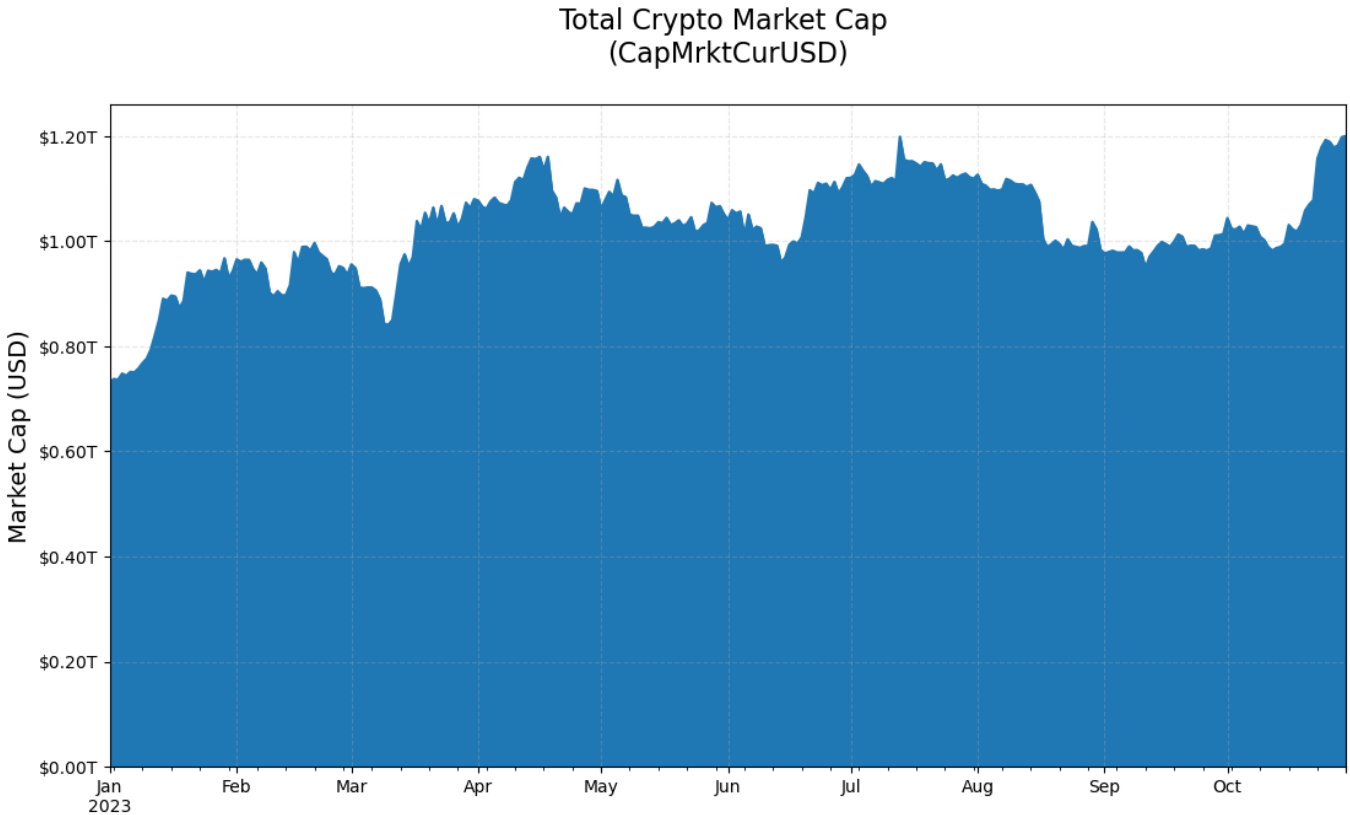
asset	1inch	aave	ada	algo	alpha	ant	bal
time							
2023-01-01 00:00:00+00:00	578851371.621414	831948979.47479	8451539587.658916	1768591937.02466	64544443.735012	87115766.669661	284339428.641511
2023-01-02 00:00:00+00:00	593808264.113895	851020271.355582	8597584057.884974	1808468934.79715	70578964.574366	82316246.224697	288403199.813244
2023-01-03 00:00:00+00:00	578865051.831563	848435934.094928	8554974393.535182	1836672352.97871	78137368.539352	82452389.917058	289756753.96142
2023-01-04 00:00:00+00:00	593238406.963806	907987443.290126	9067043606.220013	1868734277.1305	79853965.132896	86452798.779089	297519554.507435
2023-01-05 00:00:00+00:00	582087444.014613	885704860.022093	9109700818.250706	1830604723.51768	76922560.518855	84904373.55696	293385778.307155
...	...	...	...	...	...	...	...
2023-10-26 00:00:00+00:00	428758937.35098	1293789577.201675	9935348487.08849	1011964277.19008	77252911.969687	189469761.803709	204740214.601617
2023-10-27 00:00:00+00:00	420576123.60027	1254473328.869106	9992348810.910082	992285986.65516	75184551.958849	191291570.850829	198583226.846729
2023-10-28 00:00:00+00:00	430135311.575261	1280765434.383344	10053926057.215488	1017407965.64103	77903771.190297	195310940.172322	202613027.942235
2023-10-29 00:00:00+00:00	444308400.766041	1355153928.941122	10218026689.446142	1064042315.8603	79403502.720731	193362544.04287	206749657.030181
2023-10-30 00:00:00+00:00	445644672.2356	1340224606.561725	10463214977.218937	1121285470.67331	81006701.580754	189592117.843778	208148105.50576

303 rows × 90 columns

In [12]:
```
current_market_cap_last = capmrktcur_pivot['Total Cap'][-1]

formatted_market_cap = '${:,.2f}'.format(current_market_cap_last)

print('Current Market Cap based on verified on-chain supply: ' + formatted_market_cap)
```

```
Current Market Cap based on verified on-chain supply: $1,200,280,424,649.30
/var/folders/c9/6xz4c5l97zjcmqsbt1gqb9hc0000gn/T/ipykernel_82239/2276337715.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as labels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  current_market_cap_last = capmrktcur_pivot['Total Cap'][-1]
```

In [13]:
```
plt.figure(figsize=(13, 7))
capmrktcur_pivot['Total Cap'].plot(kind='area', stacked=True)

# Set the title and labels
plt.title('Total Crypto Market Cap \n(CapMrktCurUSD)\n',fontsize=16)
plt.xlabel('')
plt.ylabel('Market Cap (USD)',fontsize=14)
plt.grid(True, alpha=0.3, linestyle='--')
formatter = mticker.FuncFormatter(lambda x, pos: '${:,.2f}T'.format(x/1000000000000))
plt.gca().yaxis.set_major_formatter(formatter)

plt.show()
```



Estimated Market Cap

Due to the complexity of running blockchain nodes, some assets pose additional challenges in directly verifying supply data. Coin Metrics partners with CoinGecko to offer a "estimated supply" metric, sourced from a variety of third-party sources like token projects or blockchain explorers.

The **CapMrktEstUSD** metric offers slightly wider asset coverage, with the trade-off of leveraging more "trusted" sources for supply.

In [14]:
```
catalog_est = client.catalog_asset_metrics(metrics='CapMrktEstUSD').to_dataframe()
```

In [15]:
```
catalog_est
```

Out[15]:

	metric	full_name	description	product	category	subcategory	unit	data_type	type	display_name	frequency	asset
0	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	1inch
1	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	aave
2	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	aca
3	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	ach
4	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	ada
...	...	...	...	...	...	...	...	...	...	...	...	...
381	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	zec
382	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	zen
383	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	zil
384	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	zks
385	CapMrktEstUSD	Capitalization, market, estimated supply, USD	The sum USD value of the estimated supply in c...	Network Data	Market	Market Capitalization	USD	decimal	Product	Market Cap Estimated (USD)	1d	zrx

386 rows × 12 columns

In [16]:
```
est_assets = catalog_est['asset'].to_list()
```

In [17]:
```
capmrktest = client.get_asset_metrics(
    assets=est_assets,
    metrics='CapMrktEstUSD',
    start_time='2023-01-01'
).to_dataframe()
```

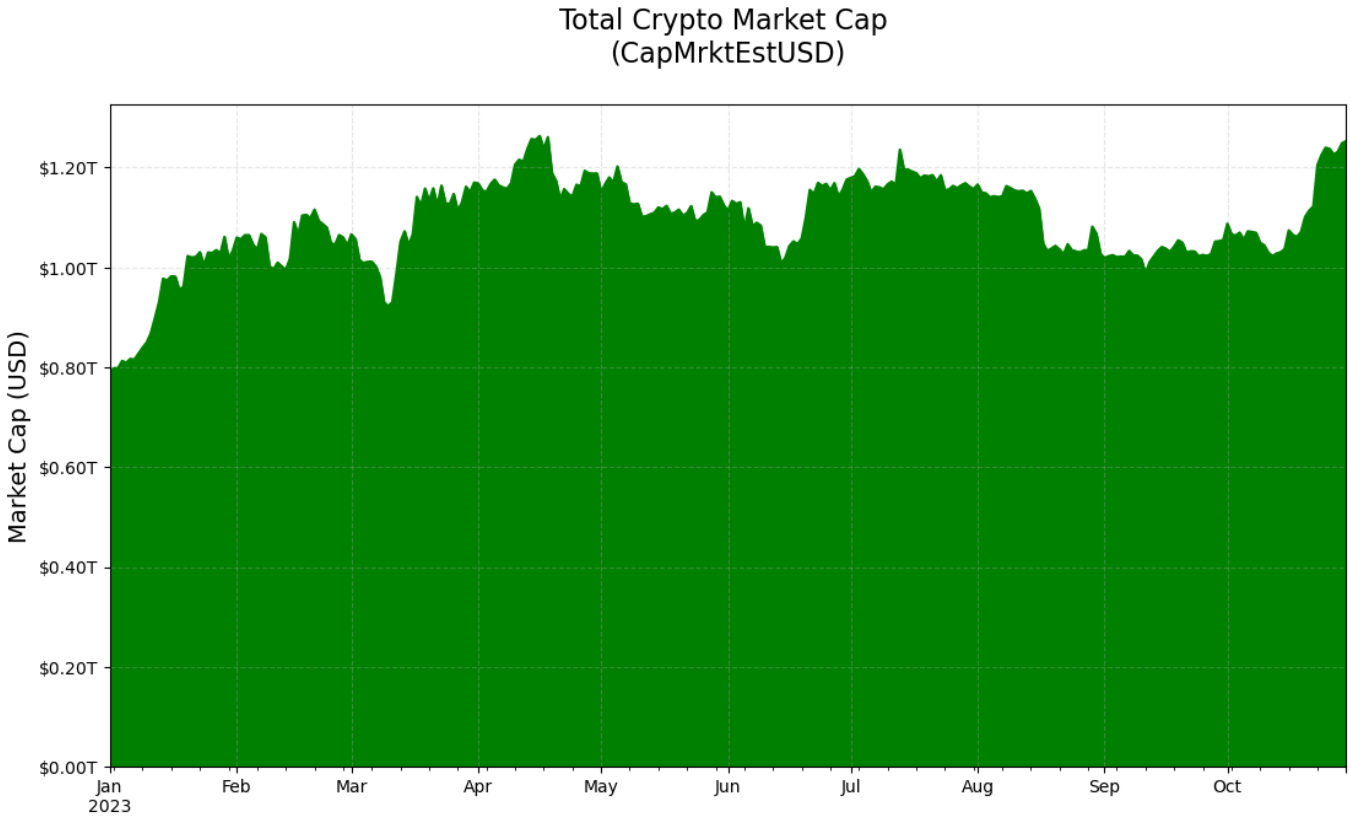
In [18]:
```
capmrktest_pivot = capmrktest.pivot(index='time',columns='asset',values='CapMrktEstUSD')
```

In [19]:
```
capmrktest_pivot['Total Cap'] = capmrktest_pivot.sum(axis=1)
```

In [20]:
```
est_market_cap_last = capmrktest_pivot['Total Cap'][-1]

formatted_est_market_cap = '${:,.2f}'.format(est_market_cap_last)

print('Estimated Market Cap based on estimated supply (3rd-party sources): ' + formatted_est_market_cap)
```

```
Estimated Market Cap based on estimated supply (3rd-party sources): $1,252,876,784,825.73
/var/folders/c9/6xz4c5l97zjcmqsbt1gqb9hc0000gn/T/ipykernel_82239/2571402615.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as labels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  est_market_cap_last = capmrktest_pivot['Total Cap'][-1]
```

In [30]:
```
plt.figure(figsize=(13, 7))

capmrktest_pivot['Total Cap'].plot(kind='area', stacked=True, color='green')
plt.title('Total Crypto Market Cap \n(CapMrktEstUSD)\n',fontsize=16)
plt.xlabel('')
plt.ylabel('Market Cap (USD)',fontsize=14)
plt.grid(True, alpha=0.3, linestyle='--')

formatter = mticker.FuncFormatter(lambda x, pos: '${:,.2f}T'.format(x/1000000000000))
plt.gca().yaxis.set_major_formatter(formatter)

plt.show()
```



Free Float Market Cap

While traditional market capitalization metrics rely on circulating supply to quantify the aggregate value of an asset, circulating supply may not necessarily be the appropriate metric for measuring the liquid, readily-available count of units available on the markert.

Free Float Market Capitalization, or **CapMrktFFUSD**, is a measure of the market value of an asset's supply that is issued and available to market participants. This excludes supply that is held by insiders (i.e. protocol treasuries), controlling investors, and long term strategic holders (units with 5+ years of inactivity).

In [22]:
```
catalog_ff = client.catalog_asset_metrics(metrics='CapMrktFFUSD').to_dataframe()
```

In [23]:
```
catalog_ff
```

Out[23]:

	metric	full_name	description	product	category	subcategory	unit	data_type	type	display_name	frequency	asset
0	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	1inch
1	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	aave
2	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	ada
3	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	algo
4	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	alpha
...	...	...	...	...	...	...	...	...	...	...	...	...
86	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	xtz
87	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	xvg
88	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	yfi
89	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	zec
90	CapMrktFFUSD	Capitalization, market, free float, USD	The sum USD value of the current free float su...	Network Data	Market	Market Capitalization	USD	decimal	Product	Free Float Market Cap (USD)	1d	zrx

91 rows × 12 columns

Comparing BTC market capitalization with BTC free float market capitalization

In [24]:
```
btc_ff_and_cur = client.get_asset_metrics(
    assets='btc',
    metrics=['CapMrktFFUSD','CapMrktCurUSD'],
    start_time='2020-01-01'
).to_dataframe()
```

In [25]:
```
btc_ff_and_cur
```

Out[25]:

	asset	time	CapMrktCurUSD	CapMrktFFUSD
0	btc	2020-01-01 00:00:00+00:00	130044373322.333786	101631140309.664062
1	btc	2020-01-02 00:00:00+00:00	125997729470.887527	98470895441.043961
2	btc	2020-01-03 00:00:00+00:00	132696546617.941193	103706959483.227219
3	btc	2020-01-04 00:00:00+00:00	133217241900.653427	104115224222.326797
4	btc	2020-01-05 00:00:00+00:00	133275140628.500854	104158964372.081024
...	...	...	...	...
1394	btc	2023-10-26 00:00:00+00:00	667384425419.412842	468917250020.46051
1395	btc	2023-10-27 00:00:00+00:00	661861216442.481689	465102737944.660828
1396	btc	2023-10-28 00:00:00+00:00	665695994028.704346	467848845580.879456
1397	btc	2023-10-29 00:00:00+00:00	675152083745.63855	474502062292.43927
1398	btc	2023-10-30 00:00:00+00:00	673638496021.274902	473443828871.129578

1399 rows × 4 columns

In [31]:
```
plt.figure(figsize=(13, 7))

plt.plot(btc_ff_and_cur['time'], btc_ff_and_cur['CapMrktCurUSD'] / 1e9, label='Market Cap', color='blue')
plt.plot(btc_ff_and_cur['time'], btc_ff_and_cur['CapMrktFFUSD'] / 1e9, label='Free Float Market Cap', color='green')
```

```
plt.title('Bitcoin Market Cap Over Time\n(Circulating vs. Free Float Supply)\n', fontsize=16)
plt.xlabel('')
plt.ylabel('Market Cap (USD)', fontsize=14)
plt.grid(True, alpha=0.3, linestyle='--')

formatter = mticker.FuncFormatter(lambda x, pos: '${:,.0f}B'.format(x))
plt.gca().yaxis.set_major_formatter(formatter)
plt.gca().xaxis.set_major_locator(mdates.AutoDateLocator())
plt.gca().xaxis.set_major_formatter(mdates.ConciseDateFormatter(mdates.AutoDateLocator()))

plt.legend()

plt.show()
```