



Blockchain programming

ESILV 2023/2024



Ordre du jour



APIs



Crypto currency exchanges



Common trading strategies



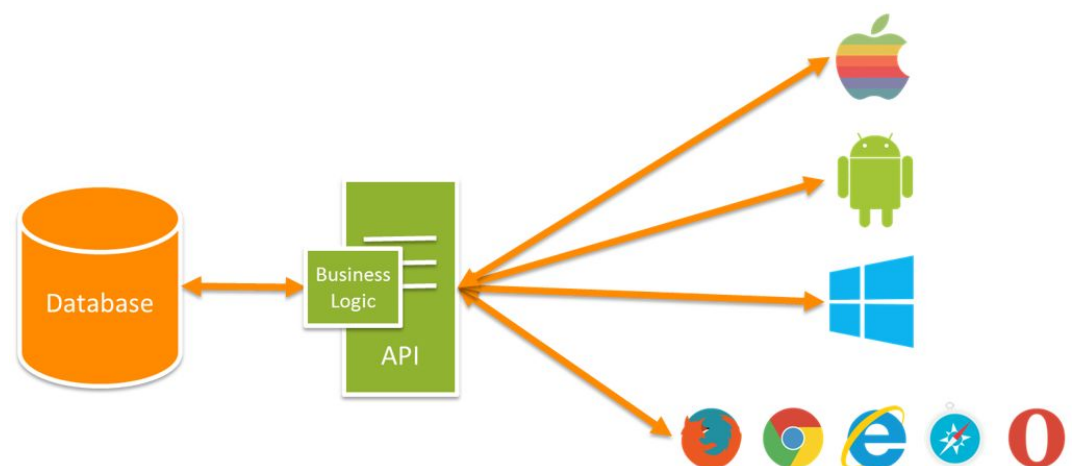
Using crypto exchange API



APIs

Application Programmable Interfaces

- Traditional web design is
backend <-> Frontend <-> Browser
- Very powerful for human centric web
- Hard to automate interactions between
machines
- API make it easier to automate processes
between different web resources



Application Programmable Interfaces

- SOAP: Simple Object Access Protocol. Very strictly defined protocol, based on XML
- Websocket: Mostly read only data streams, faster than other methods. Much less implemented
- RESTful: Representational State Transfer. Based on HTTP, less strictly defined, uses JSON a lot
- Methods:
 - GET: Retrieve resources
 - POST: Send resources
 - PUT: Modify resources
 - DELETE: Delete resources
 - Custom methods...



Crypto currency exchanges

The logo for Poloniex, featuring the word "POLONIEX" in a bold, teal-colored, sans-serif font.

Crypto exchanges

The logo for Binance, featuring a stylized orange diamond shape above the word "BINANCE" in a bold, orange, sans-serif font.The logo for Paymium, featuring a blue square above the word "PAYMIUM" in a bold, black, sans-serif font, with the tagline "MONEY OVER IP" in a smaller, black, sans-serif font below it.The logo for GDAX, featuring three vertical blue bars of increasing height to the left of the word "GDAX" in a bold, black, sans-serif font.

- There are hundreds of crypto exchanges
- When programming robots, points of attention:
 - Is the exchange safe? (risk of loss of funds)
 - Is the volume worth it?
 - What cryptos are traded on it?
 - How stable is the API?
 - How well documented is the API?

The logo for Bitstamp, featuring the word "BITSTAMP" in a bold, sans-serif font, with "BIT" in green and "STAMP" in black.The logo for Kraken, featuring a stylized blue Kraken head above the word "kraken" in a bold, black, sans-serif font.

Common indicators

- Bid: The most someone is willing to pay for an asset
- Ask: The least somebody is willing to receive for an asset
- Candles: Bundles of transactions
 - Duration: Length in time of the candle
 - Open: First transaction in the candle
 - Close: Last transaction
 - High/Low: extreme price values of the candle



Common trading strategies

Swing trading

- Trying to predict future movements of the market
- Buying/Selling depending on expected outcomes
- Useful tools:
 - Data collection APIs
 - Analytics
 - Order book management

Arbitrage

- Trying to gain from market inefficiency
- EG: Buying BTC at 1000 euros on exchange A, selling at the same time for 1005 euros on exchange B
- Requires various API libraries
- Requires pools of funds spread around
- Real time is key



Using crypto exchanges APIs

Using APIs

Prerequisites

- Use Python 3 or javascript
- No precompiled trading module, write the REST calls yourself using a rest module
- Use Binance API (or Coinbase if you prefer)
- Create a function for each task
- Do not store your credentials on your github!

Tasks list - GET

- Create a git repository and share it with the teacher (0xEniotna, make it public please)
- Get a list of all available cryptocurrencies and display it
- Create a function to display the 'ask' or 'bid' price of an asset. Direction and asset name as parameters

```
def getDepth(direction='ask', pair = 'BTCUSD')
```
- Get order book for an asset

Using APIs

Tasks list – GET

- Create a function to read aggregated trading data (candles)
`def refreshDataCandle(pair = 'BTCUSD', duration = '5m')`
- Create a sqlite table to store said data (schema attached in the next slide)
- Store candle data in the db
- Modify `refreshDataCandle()` to update when new candle data is available
- Create a function to extract all available trade data
`def refreshData(pair = 'BTCUSD')`
- Store the data in sqlite

Tasks list – POST

- Create an order
`def createOrder(api_key, secret_key, direction, price, amount, pair = 'BTCUSD_d', orderType = 'LimitOrder')`
- Cancel an order
`def cancelOrder(api_key, secret_key, uuid)`

Sqlite schema

Data candles:

```
CREATE TABLE my_table (Id INTEGER PRIMARY KEY,  
date INT, high REAL, low REAL, open REAL, close  
REAL, volume REAL)
```

Full data set:

```
CREATE TABLE my_table(Id INTEGER PRIMARY KEY,  
uuid TEXT, traded_crypto REAL, price REAL,  
created_at_int INT, side TEXT)
```

Keeping track of updates:

```
CREATE TABLE last_checks(Id INTEGER PRIMARY KEY,  
exchange TEXT, trading_pair TEXT, duration TEXT,  
table_name TEXT, last_check INT, startdate INT, last_id  
INT);
```

References

Wikipedia page for APIs

https://fr.wikipedia.org/wiki/Interface_de_programmation

Using requests in Python

<https://www.pythonforbeginners.com/requests/using-requests-in-python>

Binance API Documentation

<https://github.com/binance/binance-spot-api-docs/blob/master/rest-api.md>

Coinbase pro API documentation

<https://docs.pro.coinbase.com/>

Thank you

For your attention !

