User Manual

api_trading.py
09 May 2023, 09:27 am

1 Share registered phone number / email and static IP address

2 Steps to get static IP address :work-in-progress:

3 Receive API key and Secret key via encrypted mail :work-in-progress:

4 Steps to access encrypted mail :work-in-progress:

5 Steps to setup the Signature Creation script on your system

- Download the script from mail and save in your system
- Download VSCode on your system
- Click on the mentioned icon on left panel in VSCode and search for `Python extension` and install it

For Mac users:

- Open terminal using the command `Ctrl + ``
- Set up virtual environment to run python script via :
  `python3 -m venv venv`
  `source venv/bin/activate`
- Upgrade pip to install required libraries
  `pip install --upgrade pip`
- Install required libraries
  `pip install cryptography==40.0.2`
  `pip install requests==2.27.1`
- Run the python script
  `python <file_name_on_system>.py`

For Windows users:

- Open terminal using the command `Ctrl + ``
- Set up virtual environment to run python script via :
  `pip install virtualenv`
  `virtualenv myenv`
  `myenv\Scripts\activate`

- Upgrade pip to install required libraries
  `pip install --upgrade pip`
- Install required libraries
  `pip install cryptography==40.0.2`
  `pip install requests==2.27.1`
- Run the python script
  `python <file_name_on_system>.py`

> ⚠️ Incase an internal user, please connect to VPN before hitting any API via script

6️⃣ Steps to check the script for API Trading

- Add your `secret_key` and `api_key` received in mail in the given script in the manner represented below:

```
secret_key = "<secret_key received in mail>"
api_key = "<api_key received in mail>"
api_trading_client = ApiTradingClient(secret_key, api_key)
```

- Run the following code via `python <file_name_on_system.py>` to ensure if successfully able to connect to server:

```
#check connection
print(api_trading_client.check_connection())
```

- If success ✅ response received, you are good to start with API Trading on CoinSwitch Pro 🥳

7️⃣ Using Script for API Trading on CoinSwitch Pro

▶️ <u>Create Order</u>

1. Add the values in given payload as per requirement and run the script with `python <file_name_on_system>.py`

```
#create order
payload = {
  "deposit_amount": 160,
  "deposit_currency": "inr",
  "destination_currency": "btc",
  "expiry_period": 90,
  "limit_price": 2200000,
  "type": "LIMIT",
  "exchange_name": "WAZIRX",
  "uuid": "<random and unique string of characters and numbers>"
}
print(api_trading_client.create_order(payload=payload))
```

2. ***Note:*** `uuid` needs to be changed for each order placed
3. ***Note:*** All parameters mentioned are mandatory
4. Allowed values for `exchange_name` : `CSX`, `WAZIRX`
5. `deposit_currency` and `destination_currency` vary as per order `trade_type` like `buy/sell`
6. `deposit_amount` should be according to expected precision of `coin pair` and `exchange`. You can use the following payload to run the script with `python <file_name_on_system>.py` and get exchange precision values for coin pairs
7.

```
# get exchange precision
payload = {
```

```
    "exchange_name":"csx"
}
print(api_trading_client.get_exchange_precision(payload = payload))
```

8. Allowed values for `exchange_name` : `csx` , `wx`

▶️ Cancel Order

1. Add the `order_id` of the order to be cancelled in given payload and run the script with `python <file_name_on_system>.py`
2. ***Note:*** All parameters mentioned are mandatory

```
#cancel order
payload = {
    "order_id": "<order_id>",
    "action": "DELETE"
}
print(api_trading_client.cancel_order(payload=payload))
```

▶️ Get Portfolio

1. To get portfolio , run the script with given code snippet using `python <file_name_on_system>.py`

```
#get portfolio
print(api_trading_client.get_user_portfolio())
```

▶️ Get All Open Orders

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```
#get open orders
params = {
  "page": 1,
  "count": 1,
  "from_date": "2022-1-1",
  "to_date": "2023-5-4",
  "trade_type": "sell",
  "currency": '''["btc,inr"]''',
  "exchange": '''["csx","wx"]''',
  "order_type": "LIMIT"
}
print(api_trading_client.get_open_orders(params = params))
```

2. ***Note:*** All parameters mentioned above are optional incase you want list of all open orders
3. Allowed values for `trade_type` : `buy` , `sell`
4. Ensure `currency` value is of type `["btc,inr", "eth,inr", "doge,inr"]`
5. Allowed values for `exchange` : `["csx"]` , `["wx"]` , `["csx", "wx"]`

▶️ Get All Closed Orders

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```
#get closed orders
params = {
  "page": 1,
  "count": 1,
```

```
  "from_date": "2022-1-1",
  "to_date": "2023-5-4",
  "trade_type": "sell",
  "currency": '''["btc,inr"]''',
  "exchange": '''["csx","wx"]''',
  "order_type": "LIMIT"
}
print(api_trading_client.get_closed_orders(params = params))
```

2. ***Note:*** All parameters mentioned above are optional incase you want list of all closed orders
3. Allowed values for `trade_type` : `buy`, `sell`
4. Ensure `currency` value is of type `["btc,inr", "eth,inr", "doge,inr"]`
5. Allowed values for `exchange` : `["csx"]`, `["wx"]`, `["csx", "wx"]`

▶️ Get 24hr data of all coin pairs on an exchange

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```
params = {
  "exchange": 'csx'
}
print(api_trading_client.get_24h_all_pairs_data(params=params))
```

2. ***Note:*** All parameters mentioned above are mandatory
3. Allowed values for `exchange` : `"csx"`, `"wx"`

▶️ Get 24hr data of a coin pair on all exchanges

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```
params = {
  "symbol": "btc,inr",
  "exchange": '["csx"]'
}
print(api_trading_client.get_24h_coin_pair_data(params=params))
```

2. ***Note:*** All parameters mentioned above are mandatory
3. Allowed values for `exchange` : `"csx"`, `"wx"`, `["csx", "wx"]`

▶️ Get 24hr data of a coin pair on all exchanges

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```
params = {
  "symbol": "btc,inr",
  "exchange": '["csx"]'
}
print(api_trading_client.get_24h_coin_pair_data(params=params))
```

2. ***Note:*** All parameters mentioned above are mandatory
3. Allowed values for `exchange` : `"csx"`, `"wx"`, `["csx", "wx"]`

▶️ Get Candlestick data

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```python
# get candelstick data
params = {
  "to_time": "1662681600000",
  "from_time": "1647388800000",
  "symbol": "BTCINR",
  "c_duration": 1440,
  "exchange": "wx"
}
print(api_trading_client.get_candelstick_data(params=params))
```

2. ***Note:*** All parameters mentioned above are mandatory
3. Allowed format for `symbol` : `"BTCINR"`
4. Allowed values for `exchange` : `"csx"`, `"wx"`
5. `from_time` and `to_time` are epoch timestamps, refer: Epoch Converter

▶ Get Depth

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```python
#get depth
params = {
  "exchange": "csx",
  "symbol": "btc,inr"
}
print(api_trading_client.get_depth(params = params))
```

2. ***Note:*** All parameters mentioned above are mandatory
3. Allowed format for `symbol` : `"btc,inr"`
4. Allowed values for `exchange` : `"csx"`, `"wx"`

▶ Get Trades

1. Add the values in given params as per requirement and run the script with `python <file_name_on_system>.py`

```python
# get trades
params = {
  "exchange": "csx",
  "symbol": "btc,inr"
}
print(api_trading_client.get_trades(params=params))
```

2. ***Note:*** All parameters mentioned above are mandatory
3. Allowed format for `symbol` : `"btc,inr"`
4. Allowed values for `exchange` : `"csx"`, `"wx"`