GCEX WebSocket API

GCEX Websocket Interface

Version 1.0

Introduction

GCEX offers a Websocket interface to clients who wish to interact with GCEX over API and who do not want to use the FIX Interface.

This document will describe API, message types and message flows. All clients are encouraged to thoroughly test their application against the UAT endpoints before using in a production environment.

Connectivity

The sessions operate on a one-to-one basis meaning each session can only be logged into by one client application at a given time. In order to access the Websocket endpoint, it is required to supply source IP range(s).

All Websocket connections are secured with TLS and routed through the internet, cross connect is not available.

Endpoint

The endpoint required Basic HTTP Authentication, details will be provided during onboarding.

Message Types

Message Reject (server->client)

Example

```
{
  "msg_type": "market_data_request",
  "errors": [ "symbol must not be empty" ],
  "original_message": "{\"type\":\"market_data_request\"}",
  "type": "message_reject"
}
```

Properties

- type: message_reject
- msg_type: message type of the original message
- errors: list of errors
- original_message: original message string

Message Data Request

Example

```
{
  "type": "market_data_request",
  "symbol": "EUR/USD"
}
```

Properties

- type: market_data_request
- symbol

Message Data Snapshot (server->client)

Example

```
{
  "symbol": "BTC/USD",
  "bids": [ { "price": 28295, "amount": 7.5 }, { "price": 28292, "amount": 10 } ],
  "asks": [ { "price": 28331, "amount": 7.5 }, { "price": 28334, "amount": 10 } ],
  "timestamp": "2023-03-29T13:12:18.505Z",
  "type": "market_data_snapshot"
}
```

Properties

- type: market_data_snapshot
- symbol
- bids
- asks
- timestamp: time in ISO-8601 UTC,

Market Data Reject (server->client)

Example

```
{ "symbol": "CFD/BTC/USD", "type": "market_data_reject" }
```

Order (client->server)

Example

```
{
  "type": "order",
  "symbol": "BTC/USD",
  "account": "AGL_TEST_02",
  "client_order_id": "953b4970-fab9-4b47-8387-bcece3d78f45",
  "order_type": "LIMIT",
  "side": "BUY",
  "time_in_force": "FILL_OR_KILL",
  "price": 1.00,
  "amount": 1.00
}
```

Properties

- type: order
- symbol

- account
- client_order_id: unique id, max 100 characters, must contain only `0-9`, `A-z`, `-`
- order_type:
  - LIMIT
  - MARKET
  - STOP_LIMIT
  - STOP_MARKET
- side:
  - BUY
  - SELL
- time_in_force:
  - FILL_OR_KILL
  - IMMEDIATE_OR_CANCEL
  - DAY
  - GOOD_TILL_CANCEL
- price: only if order_type is LIMIT or STOP_LIMIT
- stop_price: only if order_type is STOP_MARKET or STOP_LIMIT
- amount

Execution (server->client)

Example

```
{
  "execution_id": "1375513756527759360",
  "order_id": "1375513755982499840",
  "client_order_id": "953b4970-fab9-4b47-8387-bcece3d78f45",
  "side": "BUY",
  "symbol": "BTC/USD",
  "quantity": 0,
  "price": 0,
  "average_price": 0,
  "cumulative_quantity": 0,
  "total_quantity": 1,
  "leaves_quantity": 0,
  "order_status": "CANCELED",
  "execution_type": "CANCELED",
  "order_type": "LIMIT",
  "time_in_force": "FILL_OR_KILL",
  "timestamp": "2023-03-29T13:12:18.505Z",
  "reason": "Order Canceled",
  "type": "order_execution"
}
```

Properties

- execution_id: id identifying this execution,
- order_id: id identifying order,
- client_order_id: client order id as provided in the original order,
- side: side as provided in original order,
- symbol: symbol as provided in original order,
- quantity: quantity of this execution,
- price: price of this execution,
- average_price: average price of cumulative quantity,

- cumulative_quantity: cumulative quantity,
- total_quantity: quantity as provided in original order,
- leaves_quantity: quantity left to fill,
- order_status: order status at this execution,
  - PENDING_NEW
  - NEW
  - PART_FILLED
  - PENDING_CANCEL
  - PENDING_AMEND
  - SUSPENDED
  - FILLED
  - CANCELED
  - REJECTED
  - EXPIRED
- execution_type: type of this execution,
  - NEW
  - CANCELED
  - REPLACED
  - PENDING_CANCEL
  - REJECTED
  - PENDING_AMEND
  - TRADE
  - TRADE_CORRECT
  - TRADE_CANCEL
  - TRADE_RELEASED_CLEARING
- order_type: order type as provided in original order,
- time_in_force: time in force as provided in original order,
- timestamp: time in ISO-8601 UTC,
- reason: optional reason,
- type: type of this message

The order status reflects the current status of the order. It should follow the state changes as specified in the FIX protocol (see here).

Typical flow is as follows:

1. PENDING_NEW (Might be omitted)
2. NEW
3. PART_FILLED (Might not occur) | PENDING_CANCEL (Might not occur) | SUSPENDED (Might not occur)
4. FILLED | REJECTED | CANCELLED | EXPIRED (FINAL State)

The execution type reflects the type of this execution, note that this can result in the following order_status and execution_type:

- PENDING_CANCEL | TRADE
- PENDING_AMEND | TRADE

Order Cancel (client->server)

Example


```
{
  "type": "order_cancel",
  "symbol": "ETH/USD",
  "account": "AGL_TEST_02",
  "cancel_id": "2b39bd4d-6639-491d-883a-580cd4cc44d4",
  "client_order_id": "e4e88791-fea7-4c8e-be42-8e0ed39a4e8d",
  "side": "BUY",
  "amount": 0.46,
  "order_id": "1375813192080429056"
}
```


Properties

- type: order_cancel,
- symbol: symbol as provided in original order,
- account: AGL_TEST_02,
- cancel_id: unique order cancellation id,
- client_order_id: client order id as provided in original order,
- side: side as provided in original order,
- amount: amount as provided in original order,
- order_id: order id as received in execution for order with `client_order_id`

When the order cancellation takes place, you will receive an execution with the order id and a `client_order_id` equal to `cancel_id`. You will have to match it to the original order based on `order_id`.

Example Cancelation Execution


```
{
  "execution_id": "1375894563922644992",
  "order_id": "1375894479919124480",
  "client_order_id": "c669b16b-dbb9-4716-afa8-8762ab47d534",
  "side": "SELL",
  "symbol": "BTC/USD",
  "quantity": 0,
  "price": 0,
  "average_price": 0,
  "cumulative_quantity": 0,
  "total_quantity": 6.95,
  "leaves_quantity": 0,
  "order_status": "CANCELED",
  "execution_type": "CANCELED",
  "order_type": "LIMIT",
  "time_in_force": "DAY",
  "epoch": 1679585841001,
  "reason": "Cancelled by user.",
  "type": "order_execution"
}
```


Order Cancel Reject (server->client)

Example

```
{
  "client_order_id": "55ed0e41-dde0-4ba3-9349-b3019b7b4e13",
  "order_status": "REJECTED",
  "reason": "rejected cancel for a8ad2b9c-64b5-4fab-ba64-31e4848bf9c0 due to : Can\\u0027t
       cancel order type, not passive!",
  "type": "order_cancel_reject"
}
```

Properties

- type: order_cancel_reject,
- client_order_id: client order id as provided in order cancel as `cancel_id`,
- reason: order cancellation reject reason
- order_status: order status of the order cancel

Example Client

Requires:

- https://www.npmjs.com/package/ws

Run with:

```
./client.js ws://username:password@url/ws BTC/USD
```

```
#!/usr/bin/env node
const WebSocket = require('ws');

async function sleep(msec) {
    return new Promise(resolve => setTimeout(resolve, msec));
}

async function main(ws) {
    var instrument = process.argv[3];
    console.log("Connected.");
    await new Promise(resolve => ws.once('open', resolve));
    ws.on('message', (data) => {
        var js = JSON.parse(data);
        console.log(js);
    })

    var request = { "type": "market_data_request", "symbol": instrument };
    var data = JSON.stringify(request);
    console.log("Sending " + data);
    ws.send(data);
    console.log("Sent");
    await sleep(10000);

    return ws;
}

var host = process.argv[2];
var ws = new WebSocket(host);
main(ws)
    .then(client => {
        if(client instanceof WebSocket){
            client.close();
        }
    })
    .catch(client => {
        if(client instanceof WebSocket){
            client.close();
        }
    });
```