cServer Manager's API Documentation

Spotware Systems Ltd

Version 1.0, 2022-03-16

Table Of Contents

1. Definitions 1
2. Warning before integration 2
3. Protocol description and integration package 3
4. Establishing Connection 3
5. Authorization in cServer 5
6. Trading and CRUD 5
7. ProtoExecutionEvent 6
8. Very useful information 7

1. Definitions

cServer – server-side JavaTM application, responsible for Account managing, trading, charting, price generation, etc.

cID – server-side JavaTM application, responsible for “User” entity, linkage of Accounts to Users, notifications (push/emails), authentication

Proxy – server-side JavaTM application, responsible for internal routing of commands and is the only one connection point for external clients. Setup can contain from one to unlimited amount of proxies depending on your desired geo presence. Wide Proxy cloud extremely improved client’s connectivity. Every Proxy can route requests to cServer or/and to cID and desired Proxy for connection must be selected according to the best (smallest) ping.

cBroker – client application, designed for managing, changing and reporting of all main server-side entities like Accounts, Groups, Symbols, Orders, Deals, etc.

cTMobile/cTWeb/cTDesktop Applications – client applications, developed by Spotware Systems Ltd., using different programming languages. Such application are developed in a best way of supporting server-side functionality and allow users to control their Balances, Orders, Positions, build technical analysis, etc.

Your Server – is application under your control which will be integrated with cServer using provided cServer Manager’s API. In this setup Your Server is a Client for server-side products.

Your Client Applications – any kind of software you control, which acts like a Client to Your Server. Manager’s API – disclosed list of requests/responses/events, which might be sent/received by Your Server to cServer or cID (via Proxy) for the purposes of current integration. Note that current version of Manager’s API support only cServer related commands; communication with cID is not supported at the moment.

2. Warning before integration

Starting from this moment cServer products became part of your business. And we hope that it will grow. That's why we'd like to warn you before you start integration: don't ruin server and as result – your business.

Manager's API is a trusted protocol. It means we allow a lot of things to be done through it. And we rely on understanding of actions, requested via Manager's API, by you.

There are several ways of solving one business task via Manager's API. But some of them are good, some – bad. Bad ways can significantly affect performance of server-side products (and even crash them) and that might affect your business negatively. So, please try to avoid it.

Example #1:

Business task: always have actual trader's balances into Your Server.

Bad way: every second send to cServer request, which will return list of all Accounts with their balances.

Correct way: handle Execution Events from cServer, which are sent every time when Account balance is changed.

Example #2:

Business task: keep list of all Accounts' Trades

Bad way: periodically or on restart request all Trade history of all traders from 1970 to 2038 years.

Correct way: initially download the history from 1970 to current date, persist it into DB/Disk. On restarts/reconnections – request Trades from last known deal to current date. During the runtime – handle execution events instead of historical requests.

We believe that Manager's API contain full list of commands, needed for you and its structure allows to have smooth and correct logic on your side. If you think that you need something else or you can't decide and find the best way of solving your technical or business cases, related to our Manager's API, please do not hesitate to contact us at support@spotware.com. We'll be happy to assist you and find the best way of using Manager's API or will advise you some applicable workaround or develop some new functionality for you.

3. Protocol description and integration package

Manager's API is based on Google Protocol Buffers V2 (proto2). Overview of the protocol processing you can find at official website https://developers.google.com/protocol-buffers/. Additional example is given in the next section.

Encoding on the wire is also described in cTrader OpenAPI page, which intentionally has the same way of connection: https://spotware.github.io/open-api-docs/protocol-buffers/

Integration package contain:

1. Current API documentation
2. 4 .proto files:
    - CSModelMessages_External.proto – contains list of cServer entities
    - CSMessages_External.proto – contains list of cServer requests/responses/events
    - CommonModelMessages_External.proto – contains list of entities, common for cServer and cID
    - CommonMessages_External.proto – contains list of requests/responses/events, applicable for cServer and cID

4. Establishing Connection

As mentioned above, Proxy – is the connection point to all client types.

Choose any Proxy from the list, provided by our support team and establish SSL connection to it.

We support multiple connections under the same credentials, but you must create new connections for each session (and remember about bad ways).

As soon as connection is established you will receive ProtoHelloEvent and you can start Authorization process.

Here is how the TCP packet's payload looks like which contains ProtoHelloEvent:

```
+-------------------------------------------------------------------------+
|         0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f                  |
+--------+-----------------------------------------------+----------------+
|00000000|00 00 00 05|08 de 07 12 00|                   |.........       |
         |  length   |    payload   |
+--------+-----------------------------------------------+----------------+
```

- First 4 bytes `00 00 00 05` signify the amount of bytes in actual payload that comes afterwards, = 5 bytes here. You should read that amount of consequent bytes as byte array.

- Actual payload is always wrapped into ProtoMessage, which has the following structure:

```
message ProtoMessage {
    required uint32 payloadType = 1; // Contains ID of the ProtoPayloadType or other
custom PayloadTypes (e.g. ProtoCHPayloadType).
    optional bytes payload = 2; // Serialized protobuf message that corresponds to the
payloadType
    optional string clientMsgId = 3; // The request message ID, assigned by the client
that will be returned in the response.
}
```

- First field `payloadType` determines which exact message is sent/received inside `payload` field: `08` shows that it's that field that is encoded and `de 07` equals to `payloadType` of ProtoHelloEvent (`PROTO_HELLO_EVENT = 990`, subject to Base 128 Varints encoding)
- Next byte `12` shows that following bytes are actual `payload` field, which is empty as ProtoHelloEvent has no fields: `00`.
- Please note system architecture is little-endian (little end first), you must reverse the byte array you want to read or write.

5. Authorization in cServer

Authorization is done using the same credentials as for login to cBroker. As result, list of available API commands depends on the access right of the authorized Manager.

1. Into established connection send ProtoManagerAuthReq
2. Server will respond either with ProtoManagerAuthRes with available permissions (Success) or with ProtoErrorRes with related errorCode and description.

6. Trading and CRUD

Manager's API allows to Create/Read/Update/Delete many entities.

CRUD operations on Account (ProtoTrader), Group (ProtoGroup) and other similar entities are done via snapshot (it means you must provide full entity into creation request). Although, operations with Orders are done via specific commands and server will generate ProtoOrder according to the parameters from initial request.
Rules for snapshot manipulations:

1. Always set ids=0 on creation of entity. For example, into ProtoCrudTraderReq with ProtoCrudOperation=PROTO_CREATE always set ProtoTrader.traderId=0
2. Parameters defined as optional are not necessary optional from the business point of view. All newly added fields are always defined as optional in order to keep backward compatibility. Server will notify you in case optional field, which is required from the business point of view is missing into request.
3. Do not try to set all optional parameters. Fill parameters which are obvious for you. If something important is absent – server will notify you.
4. Do not try to update Account's balances via CRUD commands – it will not work. Use specific command for that (ProtoChangeBalanceReq).

General rules:

1. Most of the numeric parameters have integer type. Because of this you must correctly handle values of those parameters:
   a. All prices are specified in 1/100000 of unit of a price. (e.g. price 1.23 = protocol 123000)
   b. All volumes are specified in 0.01 of a unit (e.g. volume US$ 10 = protocol 1000)
   c. All fields related to money (balance, commission, etc.) contain the amount in 10^moneyDigits (e.g. balance = 234512 with moneyDigits = 2 is 2345.12 currency units; balance = 234512 with moneyDigits = 8 is 0.00234512 currency units)
   d. If necessary contact us at support@spotware.com to get precise information regarding specific parameter
2. Most requests will result in related response AND related event, usually matched by clientRequestId.

7. ProtoExecutionEvent

This event is a core of financial notifications. Such Event is generated in below listed cases:

- Account's balance change
- Account's bonus change
- Order-related actions (Accepted, Filled, Partially Filled, Canceled, Amended, Expired)

Depending on the case, structure of ProtoExecutionEvent will be different. ProtoExecutionEvent is always per Account or per Account per Order.

Use ProtoExecutionEvent for:

- Having actual balance information
- Having actual status of all Orders/Positions
- Knowing about all Trades (ProtoDeal – is actual Trade)

8. Very useful information

- One connection supports one cServer authorization. It is not allowed to authorize twice or more into one cServer connection. For such purpose you will need to establish new connections.
- Proxy will drop connection in case of inactivity into authorized channel. In order to avoid it send ProtoHeartbeatEvent every 25 seconds into authorized channel.
- Proxy will drop connection in case outgoing message queue is bigger than predefined amount of megabytes (70 Mb at the moment). Such queue may grow as result of not reading from channel, or huge data being taken from DB (for example, ProtoManagerDealListRes which will contain data for several years of intense trading of your Users). Be careful and use requests for lists attentively.
- After establishing authorized connection you will immediately start receiving different Events from related cServer-side product (obviously, according to activity at the related product). We do not support subscribing/unsubscribing to most of the Events.
- If you believe, that some events are useless to you – just ignore them.
- You will receive Events, described into related .proto files and some Events, which are not disclosed there and which you will not be able to parse. We deliberately limited list of disclosed Events to avoid overhead on understanding of full cServer functionality and to increase speed of your integration, by keeping only really needed data for you. We advise you to handle this situation and throw away non-parsable messages to avoid exceptions in your logs.
- Don't try to use or support all parameters and entities, received from cServer-side products – use only what is needed for your business needs.
- Starting from this moment Spotware will notify you about any changes in Manager's API in advance.