Repo API

Spotware Systems Ltd

Version 2.4.0, 2017-11-10

Table Of Contents

1. Changes

Version	Date	Changes

2.2.0	2017-03-01	* added entities `DealOffset` and `DealOffsets`
		* added `BalanceHistory.source` and `BalanceHistory.externalId`
		* added `BalanceOperationType` enum values:
		`BALANCE_OPERATION_TYPE_WITHDRAW_INVESTMENT_TO_STRATEGY`
		`BALANCE_OPERATION_TYPE_DEPOSIT_FROM_INVESTOR`

2.3.0	2017-04-05	* added `Trader.frenchRisk`

2.4.0	2017-11-10	* added `BalanceOperationType` enum values:
		`BALANCE_OPERATION_TYPE_DEPOSIT_MANAGEMENT_FEE`
		`BALANCE_OPERATION_TYPE_WITHDRAW_MANAGEMENT_FEE`
		`BALANCE_OPERATION_TYPE_DEPOSIT_PERFORMANCE_FEE`
		* added `ClosePositionDetail.performanceFee`
		* added `CommissionType.COMMISSION_TYPE_4`
		* added `OrderStatus.ORDER_STATUS_RESERVED`
		* field `Order.positionEffect` is deprecated now

2. General Description

The purpose of this document is to outline the rules of engagement for interfacing with the Spotware Systems Reporting API in order to receive real-time based server events and notifications.

Repo API uses Google Protocol Buffers version 2 for message events and for snapshots.

Event – is a real-time notification about creation/modification/deletion of any important entity on the server such as Trader, Position, Symbol, etc.

Snapshot – is a list of the entities, requested by the Client. Should be used only for getting initial snapshot of the DB structure, in order to match future events with existing entity or to get some missed entity by its ID.

Optimal way of getting snapshots is to take them during weekend.

Structure of the used messages described in the ReportingMessages.proto (must be provided additionally). See https://developers.google.com/protocol-buffers/docs/proto how to work with messages and how to build messages for your favourite language. Protocol compiler you can find here: https://developers.google.com/protocol-buffers/docs/downloads

3. Connectivity

3.1. Events

For real-time events we use RabbitMQ, so in order to start receiving events there should be established connection with RabbitMQ.

Information about connectivity could be found here:

- .NET/C# Client API Guide https://www.rabbitmq.com/dotnet-api-guide.html
- Java Client API Guide https://www.rabbitmq.com/api-guide.html

Information which should be provided by Spotware admins:

- rabbit.exchange
- rabbit.host
- rabbit.port
- rabbit.username
- rabbit.password

3.2. Snapshots

For getting initial snapshots of the DB entities Spotware team should whitelist your IP address. No authentication is required.

4. Usage

4.1. Events

After connection to RabbitMQ is established you will start getting RabbitMQ messages.

These messages contain next information:

1. ContentType - should be "application/x-protobuf"
2. MessageType - type of the protocol buffers message from file ReportingMessages.proto. For example TraderGroupEvent, TraderEvent, AuthenticationEvent
3. ContentLength - size of serialized protocol buffers message in bytes.
4. Body - byte array.

So if messageType is for example TraderGroupEvent - you can restore message for Java

```
TraderGroupEvent event = TraderGroupEvent.parseFrom(message.getBody());
```

In case connection with RabbitMQ is lost, all events will be stored on our side and will be send all together as soon as connection is restored. This will help to keep you data consistent and prevent data loss to avoid additional reconciliation and main DB load.

4.2. Snapshots

It is possible to request next snapshots of entities:

1. “assets“
2. “assetClasses“
3. “authentications“
4. “balanceHistories“
5. “bonusHistories“
6. “countries“
7. “deals“
8. “orders“
9. “positions“
10. “priceSnapshots“
11. “priceStreams“
12. “symbols“
13. “symbolCategories“
14. “traders“

15. “traderGroups“
16. “swapCalculations“
17. “actions“
18. “dealOffsets”
19. “eodSpotSnapshotReports”
20. “eodOpenPositionReports”
21. “eodTraderReports”

Repo webservices are used for such purposes:

4.2.1. Get list of entities

When using this type of requests make sure that you have set a timeout no less than 20 seconds. The limit of concurrent requests can be applied (current limit is 2). Structure of the request (except “eodSpotSnapshotReports”, “eodOpenPositionReports” and “eodTraderReports” entities – see description for them below):

```
Host:port/repo/entity?id&size&direction
```

where

- `Host:port` – should be provided by Spotware
- `entity` – one of the specified above entity
- `direction` - sorting direction, can be only "asc" or "desc" (sorting by entity id). Required.
- `id` – id of the entity, positive integer value. Optional. In case of “asc” direction server will return entities with id greater than specified id. In case of “desc” direction server will return entities with id less than specified id.
- `size` – amount of returned entities, integer value from 100 to 15000 (for "traderGroups" it’s from 1 to 5). Required.

For example:

Request the first 100 Deals

```
http://{host}:{port}/repo/deals?id=0&size=100&direction=asc
or
http://{host}:{port}/repo/deals?size=100&direction=asc
```

Request 100 Deals which id are less than Deal with specified id

```
http://{host}:{port}/repo/deals?id=1001&size=100&direction=desc
```

Request the latest 100 Deals

```
http://{host}:{port}/repo/deals?size=100direction=desc
```

Structure of the request for “eodSpotSnapshotReports”, “eodOpenPositionReports” and “eodTraderReports” entities:

```
Host:port/repo/entity/date?size&page&direction
```

where:

- `Host:port` – should be provided by Spotware
- `entity` – one of the specified above entity
- `date` – the day of reports, in YYYY-MM-DD format. Required.
- `direction` – sorting direction, can be only "asc" or "desc". Required.
- `page` - page number, positive integer value. Optional.
- `size` – amount of returned entities, integer value from 100 to 15000. Required.

For example:

Request the first 100 Reports for 23/12/2016

```
http://{host}:{port}/repo/eodSpotSnapshotReports/2016-12-
23?size=100&page=0&direction=asc
```

4.2.2. Get entity by id

Request Authentication with id = 70

```
http://{host}:{port}/repo/authentications/70
```

If entity with this id doesn’t exist, it will return an error. Method is not available for “actions“, “eodSpotSnapshotReports”, “eodOpenPositionReports” and “eodTraderReports” entities. Response always has "Content-Type: application/x-protobuf". In order to build a message from received bytes you can use Authentication.parseFrom(messageBody).