Subscription API

The subscription API provides access to the transaction log, in chronological order, for bulk and real-time storage unit updates.

On this page:

- Subscribe
- Fetch Transactions

Subscribe

Provides a subscription service to the transaction log of a storage unit. The service will automatically push new transactions and the records linked with each transaction to the subscriber clients. The subscription service uses Server-Sent Events protocol for communication.

Reconnection: If the connection is lost by the client, when reconnecting it will send a **last-event-id** header with the last transaction id that was received by the client. The subscription service will resume from the **last-event-id**.

Request

The subscribe GET request receives the storage unit name and optionally the scope to subscribe to in the URL. It can also received a **startId** paramet er to start getting transactions starting on that particular one, a **doContentProcessing** (default true) boolean parameter to trigger or not the **Fetch** events from content processing modules, a **transactionsOnly** (default false) boolean parameter to indicate whether or not the record data should be sent along a add/update transaction and a **zone** parameter to indicate from which zone to get transactions from (all if omitted).

```
GET subscription/subscribe/<storage-unit>/<optional-scope>?
startId=5644f2d497cc11144081d111&doContentProcessing=true&transactionsOnly=false&zone=LOCAL_ZONE
```

Java example

```
Client client = ClientBuilder.newBuilder().register(SseFeature.class).build();
WebTarget target = client.target(new URI("http://localhost:3000/subscription/subscribe/<storage-unit-name>
/<scope>"));
EventSource eventSource = EventSource.target(target).reconnectingEvery(2, TimeUnit.SECONDS).build();
EventListener listener = new EventListener(){
  public void onEvent(InboundEvent inboundEvent) {
    System.out.println(inboundEvent.getId() + "; "
    + inboundEvent.readData(String.class));
  }
};
eventSource.register(listener);
eventSource.open();
...
eventSource.close();
```

Javascript example

```
var EventSource = require('eventsource') // https://www.npmjs.com/package/eventsource
var source = new EventSource('http://localhost:3000/subscription/subscriptionEvent/<storage-unit-name>
/<scope>')
source.onmessage = function (e) {
    console.log(e.lastEventId + ':' + e.data)
}
```

Response

The response is of type **text/event-stream** and remains open until the client decides to unsubscribe. Each transaction log entry is returned as a single event with the transaction id as the event id and the transaction JSON as the data. If the transaction has a content record associated, a second JSON object is sent with the record data as a second data entry on the response, separated from the previous with a single new line. Each new event message is separated by two new lines (\n\n).

```
id:5644f2d497cc11144081d111\n
data:{"_id" : 5644f2d497cc11144081d111, "scope" : "connector", "recordId" : "LOCAL_ZONE:
E0F57D337D5021236E353C7AB305F147", "timestamp" : 2015-11-12T20:13:08.243Z, "type" : "ADD", "__v" : 0}\n
data:{"key": "LOCAL_ZONE:E0F57D337D5021236E353C7AB305F147", "content": {"connector": {"doc": {"test":
"data"}}}}\n\n
```

Fetch Transactions

Fetch transactions in batches ordered chronologically using a startId to get the next batch of transactions.

Request

The fetch transactions GET/POST request receives the storage unit name and optionally the scope in the URL. It can also received a **startId** paramet er to start getting transactions starting on that particular one, a **doContentProcessing** (default true) boolean parameter to trigger or not the **Fetch** eve nts from content processing modules, a **transactionsOnly** (default false) boolean parameter to indicate whether or not the record data should be sent along a add/update transaction, an **includeFirst** parameter to indicate whether or not to include the transaction of **startId** or start from the next one and a **zone** parameter to indicate from which zone to get transactions from (all if omitted).

```
GET subscription/fetchTransactions/<storage-unit>/<optional-scope>?
startId=5644f2d497cc11144081d111&doContentProcessing=true&transactionsOnly=false&zone=LOCAL_ZONE
```

Response

Responds with a 200 response code. The response body contains an array of transactions and a next transaction number for the next batch call. If there are no more transactions, the next transaction number is not part of the response.

```
{"transactions":
        [
                {
                        "transaction": {
                                "_id": "578fbc689eabceb422d3b4b7",
                                "__v": 0,"fileUpdate": false,
                                "scope": "connector",
                                "recordId": "APP_ZONE:995BB4838C40E9900FD098F6205824D1",
                                "timestamp": "2016-07-20T18:01:12.234Z",
                                "type": "ADD"
                        },
                        "record": {
                                "content": {
                                        "connector": {...}
                                },
                                "key": "APP_ZONE:995BB4838C40E9900FD098F6205824D1"
                        }
                },
                {
                        "transaction": {
                                "_id": "578fbc689eabceb422d3b4b8",
                                "__v": 0,"fileUpdate": false,
                                "scope": "connector",
                                "recordId": "APP_ZONE:995BB4838C40E9900FD098F6205824D2",
                                "timestamp": "2016-07-20T18:01:15.234Z",
                                "type": "ADD"
                        },
                        "record": {
                                "content": {
                                        "connector": {...}
                                },

                                "key": "APP_ZONE:995BB4838C40E9900FD098F6205824D2"

                        }

                }
        ],
        "nextTransactionNum": "578fbc689eabceb422d3b4cb"
}
```