VisITmeta: REST-Interface

Thomas Oelsner

June 2, 2015

Contents

1 Connection Management

This section shows examples using the REST-Interface to manage connections from the dataservice to any given number of map-server. The {Connection Name} is a unique name for each connection which can be chosen freely.

1.1 Get Connections

Example Request:

```
HTTP:GET
http://example.com:8000
```

If the suffix *?onlyActive=true* is given, only active connections will be returned.

```
HTTP:GET
http://example.com:8000?onlyActive=true
```

Response:

```
["default", "exampleConn"]
```

The Response returns a JSON-Array which contains every {Connection Name} saved in the dataservice.

1.2 Save Connection

```
HTTP:PUT
http://example.com:8000/
Content-Type: application/json
{
        connectionName:{Connection Name}
        ifmapServerUrl:{map-Server},
        userName:{Username},
        userPassword:{Password}
}
```

List of required parameters:

- connectionName
- ifmapServerUrl
- userName
- userPassword

List of optional parameters:

- authenticationBasic
- truststorePath
- truststorePassword

- useConnectionAsStartup
- maxPollResultSize

Example Request:

```
HTTP:PUT
http://example.com:8000/
Content-Type: application/json
{
        connectionName: exampleConn
        ifmapServerUrl:"https://localhost:8443",
        userName:visitmeta,
        userPassword:visitmeta
}
```

Response:

```
exampleConn was saved
```

1.3 Delete Connection

Not implemented as of June 2, 2015

```
HTTP:DELETE
http://example.com:8000/{Connection Name}
```

Example Request:

```
HTTP:DELETE
http://example.com:8000/default
```

Response:

```
Not implemented
```

1.4 Connect

```
HTTP:PUT
http://example.com:8000/{Connection Name}/connect
```

Example Request:

```
HTTP:PUT
http://example.com:8000/default/connect
```

Response:

```
INFO: connecting successfully
```

1.5 Disconnect

```
HTTP:PUT
http://example.com:8000/{Connection Name}/disconnect
```

Example Request:

```
HTTP:PUT
http://example.com:8000/default/disconnect
```

Response:

```
INFO: disconnection successfully
```

2 Subscribe Management

The following section shows the handling of subscriptions. {Subscription Name} like {Connection Name} is a unique identifier which can be chosen freely.

2.1 Get Subscriptions

```
HTTP:GET
http://example.com:8000/{Connection Name}/subscribe
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/subscribe
```

Response:

```
["default", "exampleSub"]
```

2.2 Subscribe

```
HTTP:PUT
http://example.com:8000/{Connection Name}/subscribe/update
Content-Type: application/json
{
  subscribeName:{Subscription Name},
  identifierType:{Identifier Type},
  identifier:{Identifier}
}
```

Identifier Types are:

- access-request
- device
- ip-address
- mac-address

Example Request:

```
HTTP:PUT
http://example.com:8000/default/subscribe/update
Content-Type: application/json
{
  subscribeName:exampleSub,
  identifierType:device,
  identifier:exampleDevice
}
```

Response:

```
INFO: subscribe successfully
```

2.3 Delete Subscription

```
HTTP:DELETE
http://example.com:8000/{Connection Name}/
        subscribe/delete/{Subscription Name}
```

Example Request:

```
HTTP:DELETE
http://example.com:8000/default/
        subscribe/delete/exampleSub
```

Response:

```
INFO: delete subscription(exampleSub) successfully
```

2.4 Delete All Subscriptions

```
HTTP:DELETE
http://example.com:8000/{Connection Name}/
        subscribe/delete?deleteAll=true
```

Example Request:

```
HTTP:DELETE
http://example.com:8000/default/
        subscribe/delete?deleteAll=true
```

Response:

```
INFO: delete all active subscriptions successfully
```

3 Graph Management

The last sections shows how to view graphs or deltas at different timestamps.

3.1 Changes Map

```
HTTP:GET
http://example.com:8000/{Connection Name}/graph/changes
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/graph/changes
```

Response:

```
{
    "1425915295000": 1,
    "1425915342000": 1
}
```

The Response is a JSON-Object mapping timestamps on the amount of changes occurred at that time.

3.2 Initial Graph

```
HTTP:GET
http://example.com:8000/{Connection Name}/graph/initial
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/graph/initial
```

Response:

```
[{
    "timestamp": 1425915295000,
    "links": [{
        "identifiers": [{
            "typename": "device",
            "properties": {
                "name": "freeradius-pdp"
            }
        }, {
            "typename": "access-request",
            "properties": {
                "name": "ar1"
            }
        }],
        "metadata": {
            "typename": "access-request-device",
            "properties": {
                "ifmap-cardinality": "singleValue",
            }
        }
    }]
}]
```

Note: The response was reduced for an easier view.

3.3 Current Graph

```
HTTP:GET
http://example.com:8000/{Connection Name}/graph/current
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/graph/current
```

Response: See 3.2

3.4 Graph At

```
HTTP:GET
http://example.com:8000/{Connection Name}/
        graph/{Timestamp}
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/graph/1425915342000
```

Response: See 3.2

3.5 Notifies At

```
HTTP:GET
http://example.com:8000/{Connection Name}/graph/
        {Timestamp}?onlyNotifies=true
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/
        graph/314159265?onlyNotifies=true
```

Response: See 3.2. Only difference to initial, current or graph at response: each notify metadata has its own subgraph.

3.6 Delta

```
HTTP:GET
http://example.com:8000/{Connection Name}/graph/
        {Timestamp From}/{Timestamp To}
```

Example Request:

```
HTTP:GET
http://example.com:8000/default/
        graph/314159265/358979323
```

Response: See 3.2

3.7 Graph Filter

Initial, Current and GraphAt responses may be filtered. Only necessary changes are HTTP:POST instead of HTTP:GET and a Content-Type: application/json containing the filter information.

startId Identifier where the filter begins the search (represented as JSON).

maxDepth Integer value determining the maximal amount of links traveled from the first Identifier.

resultFilter Filterstring that filters Metadata. If the resultFilter is empty, no Metadata will be filtered. If the resultFilter is null, all Metadata will be filtered, resulting in a set only containing Identifiers. The filterstring should follow the filter-syntax specified by ifmap.

matchLinks Filterstring that determines what Link-types are allowed in the filtered graph. If matchLinks is empty, all Link-types are allowed. If matchLinks is null, no Link-types are allowed resulting in a Graph just containing the start Identifier.

Example Request:

```
HTTP:POST
Content-Type: application/json
http://example.com:8000/default/graph/initial
{
startId:{type:device, name:"freeradius-pdp"},
maxDepth: 10,
resultFilter:"meta:role",
matchLinks:""
}
```

Response: See 3.2