REST API Specification

Table of Contents

- Table of Contents

Overview

- Introduction

Endpoints

- Resources
  - Containers
  - Versioning
- Services
  - Backup and Restore
  - Fixity
    - HEAD Request the fixity checksum for a given digest algorithm
    - GET Get the fixity report for an object
      - Note: Default Fixity Algorithm
- Transactions
  - POST Create a new transaction
  - GET Get the current status of the repository in a transaction
  - POST Keep an existing transaction alive
  - PUT Commit an open transaction
  - DELETE Rollback and close an open transaction

Overview

Introduction

The Fedora HTTP API is generally a RESTful API. HTTP methods like GET, PUT, POST and DELETE are implemented on most resource paths. The API also relies heavily on content negotiation to deliver context-appropriate responses, and a HATEOAS-driven text/html response (providing a decent GUI experience on top of the repository).

The Fedora RDF-based responses may be serialized as:

- application/ld+json
- application/n-triples
- application/rdf+xml
- text/n3 (or text/rdf+n3)
- text/plain
- text/turtle (or application/x-turtle)

The text/html response also includes embedded RDFa markup.

Fedora implements the Linked Data Platform 1.0 Architecture, which:

> [...] describes the use of HTTP for accessing, updating, creating and deleting resources from servers that expose their resources as Linked Data. It provides clarifications and extensions of the rules of Linked Data [LINKED-DATA]:

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names
3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)
4. Include links to other URLs, so that they can discover more things

Endpoints

Resources

Repository objects can be loosely divided into two classes of resources:

- Containers ("fedora:Container"), containing RDF properties and 0 or more child resources
- Binaries, containing any binary payload (roughly corresponding to Fedora 3 datastreams)
Containers

RESTful HTTP API - Containers

Request URI: /path/to/some/resource
Methods: GET, POST, PUT, PATCH, HEAD, OPTIONS, DELETE

Versioning

RESTful HTTP API - Versioning

- Working with a versionable resource (a LDPRv)
  - GET Retrieve the versioned resource from a specific point in time.
- Working with a versions container (a LDPCv)
  - GET Get a list of the available versions of an object
  - POST Create a new versioned resource (a new LDPRm)
- Working with a versioned resource (a LDPRm)
  - GET Get a specific versioned resource

Services

Backup and Restore

Fixity

RESTful HTTP API - Fixity

Request URI: /path/to/some/binary
Methods: HEAD, GET

HEAD

Request the fixity checksum for a given digest algorithm
Checking fixity requires retrieving the content from the binary store and may take some time.

Request Headers:

WANT-DIGEST: md5, sha, sha-256, sha-512, sha-512/256

Example:
curl -I -H "Want-Digest: sha-256" "http://localhost:8080/rest/path/to/some/binary"

**Response (fixity success):**

**Status:** 200 OK

**Headers:**
- ETag: "51c5ed5ffe4b6c79233cc7573a387902f7d171e7"
- Last-Modified: Fri, 04 May 2018 18:14:47 GMT
- Content-Type: image/jpeg
- Accept-Ranges: bytes
- Content-Disposition: attachment; filename="IMG_4023.JPG.jpg"; creation-date="Fri, 04 May 2018 18:14:47 GMT"; modification-date="Fri, 04 May 2018 18:14:47 GMT"; size=47021
- Link: <http://www.w3.org/ns/ldp#Resource>;rel="type"
- Link: <http://localhost:8080/rest/examples/binary/fcr:acl>; rel="acl"
- Link: <http://localhost:8080/rest/examples/binary/fcr:metadata>; rel="describedby"
- Link: <http://localhost:8080/static/constraints/NonRDFSsourceConstraints.rdf>; rel="http://www.w3.org/ns/ldp#constrainedBy"
- Link: <http://localhost:8080/rest/examples/binary>; rel="timegate"
- Link: <http://localhost:8080/rest/examples/binary>; rel="original"
- Link: <http://localhost:8080/rest/examples/binary/fcr:versions>; rel="timemap"
- Link: <http://mementoweb.org/ns#OriginalResource>; rel="type"
- Link: <http://mementoweb.org/ns#TimeGate>; rel="type"
- Accept-External-Content-Handling: copy,redirect,proxy
- Allow: DELETE,HEAD,GET,PUT,OPTIONS
- Digest: sha-256=08692ca74dc1e08c84f30feda5e64c72617be850a7c7a5ca88b03f6cdd3db406
- Content-Length: 47021

**Status:**

- **200 OK**
- **400 Bad request** (the 'Want-Digest' request-header algorithm type is either invalid or unsupported)
- **404 Resource not found**

**Request URI:** `/path/to/some/resource/fcr:fixity`

**Methods:** GET

**Get the fixity report for an object**

Checking fixity requires retrieving the content from the binary store and may take some time. The /fcr:fixity endpoint is not part of the Fedora API Specification at this time.

**Request Headers:**

- **ACCEPT**
  - application/ld+json, application/n-triples, application/rdf+xml, application/x-turtle, text/html, text/n3, text/plain, text/rdf+n3, text/turtle

**Example:**

```
```

**Response (fixity success):**
Status: 200 OK

Headers:
Content-Type: text/turtle

Body:

@prefix premis:  <http://www.loc.gov/premis/rdf/v1#> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://localhost:8080/rest/path/to/some/resource>
  premis:hasFixity  <http://localhost:8080/rest/path/to/some/resource#fixity/1494431303920> .

<http://localhost:8080/rest/path/to/some/resource#fixity/1494431303920>
  rdf:type                 premis:Fixity ;
  rdf:type                 premis:EventOutcomeDetail ;
  premis:hasEventOutcome   "SUCCESS" ;
  premis:hasMessageDigestAlgorithm  "SHA-1" ;
  premis:hasMessageDigest  <urn:sha1:ca3392593351ef8e6554bdabfbd8bdc1002ecb6f> ;
  premis:hasSize           "1277811"^^<http://www.w3.org/2001/XMLSchema#long> .

Response (fixity failure):

Status: 200 OK
Headers:
Content-Type: text/turtle

Body:

@prefix premis:  <http://www.loc.gov/premis/rdf/v1#> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://localhost:8080/rest/path/to/some/resource>
  premis:hasFixity  <http://localhost:8080/rest/path/to/some/resource#fixity/1494445619308> .

<http://localhost:8080/rest/path/to/some/resource#fixity/1494445619308>
  rdf:type                 premis:Fixity ;
  rdf:type                 premis:EventOutcomeDetail ;
  premis:hasEventOutcome   "BAD_CHECKSUM" ;
  premis:hasEventOutcome   "BAD_SIZE" ;
  premis:hasMessageDigestAlgorithm  "SHA-1" ;
  premis:hasMessageDigest  <urn:sha1:1d3d03c656cf0a944c393bf9257c6cecd85a263f> ;
  premis:hasSize           "1287509"^^<http://www.w3.org/2001/XMLSchema#long> .

Status: 200 OK
Headers:
Content-Type: text/turtle

Body:

@prefix premis:  <http://www.loc.gov/premis/rdf/v1#> .
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://localhost:8080/rest/path/to/some/resource>
  premis:hasFixity  <http://localhost:8080/rest/path/to/some/resource#fixity/1494445619308> .

<http://localhost:8080/rest/path/to/some/resource#fixity/1494445619308>
  rdf:type                 premis:Fixity ;
  rdf:type                 premis:EventOutcomeDetail ;
  premis:hasEventOutcome   "BAD_CHECKSUM" ;
  premis:hasEventOutcome   "BAD_SIZE" ;
  premis:hasMessageDigestAlgorithm  "SHA-1" ;
  premis:hasMessageDigest  <urn:sha1:1d3d03c656cf0a944c393bf9257c6cecd85a263f> ;
  premis:hasSize           "1287509"^^<http://www.w3.org/2001/XMLSchema#long> .

Status:

<table>
<thead>
<tr>
<th>200</th>
<th>OK</th>
</tr>
</thead>
<tbody>
<tr>
<td>404</td>
<td>Resource not found</td>
</tr>
</tbody>
</table>

Note: Default Fixity Algorithm

When ingesting a binary resource, one or more checksums may be provided, see API reference (POST example 4, POST example 4b, PUT example 3). The supported algorithms are: SHA-1, SHA-256 and MD5.

By default, the algorithm used by the /fcr:fixity endpoint is SHA-1. However, that may be changed - per resource - to one of the other supported algorithms by configuring the property: fedoraconfig:defaultDigestAlgorithm. See API reference (PATCH example 2).
RESTful HTTP API - Transactions

POST Create a new transaction

After retrieving a transaction URI from the "Location" header in the response to the following HTTP request, the client can execute any POST/PUT/DELETE/GET REST API method within the transaction scope by including the transaction URI in the "Atomic-ID" request header.

Transactions are automatically closed and rolled back after 3 minutes of inactivity. Transactions can be refreshed by POSTing to the transaction URI.

Example:

```
curl -i -X POST "http://localhost:8080/rest/fcr:tx"
```

Response:

```
Status: 201 Created

Headers:
Location: http://localhost:8080/rest/fcr:tx/ce4bb2bf-8ced-4c7d-b281-f2132e3064bb
Atomic-Expires: Thu, 07 May 2020 17:34:37 GMT
```

Usage:

When a transaction has been created, it will return a Location header. Use this location (transaction URI) as the value of the "Atomic-ID" request header for performing REST API operations within the transaction. When you are done with the transaction, either commit the transaction to the repository by making a PUT request to the transaction URI, or discard the changes and rollback the transaction by making a DELETE request to the transaction URI.

For further examples of the usage of Transactions, see: Transactions

Status:

- **201** Created: if the transaction is created successfully

GET Get the current status of the repository in a transaction

To get the status of a transaction, perform a GET request on the transaction URI that was previously returned as the value of the "Location" header from the transaction creation request. The "Atomic-Expires" header indicates the time at which the transaction will auto-rollback.

Example:

```
curl -i -X GET "http://localhost:8080/rest/fcr:tx/ce4bb2bf-8ced-4c7d-b281-f2132e3064bb"
```

Response:

```
Status: 204 No Content

Headers:
Expires: Thu, 7 May 2020 17:46:17 GMT
```

Status:

- **204** No Content: If the request was successful
- **404** Not Found: Transaction not found
- **410** Gone: Transaction expired

POST Keep an existing transaction alive

By performing a POST request on an existing transaction URI, the transaction's expiration time is extended by the value of the session timeout duration (default: 3 minutes)
### Example:

```bash
curl -i -X POST "http://localhost:8080/rest/fcr:tx/ce4bb2bf-8ced-4c7d-b281-f2132e3064bb"
```

#### Response:

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>No Content: if the transaction is renewed successfully</td>
</tr>
<tr>
<td>404</td>
<td>Not Found: if the transaction doesn't exist</td>
</tr>
<tr>
<td>410</td>
<td>GONE: TRANSACTION EXPIRED</td>
</tr>
</tbody>
</table>

### PUT

**Commit an open transaction**

Any operations you made within the scope of the transaction will be applied together, meaning if any of them fail, the whole transaction will fail.

#### Example:

```bash
curl -X PUT "http://localhost:8080/rest/fcr:tx/ce4bb2bf-8ced-4c7d-b281-f2132e3064bb"
```

#### Response:

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>No Content: if the transaction is committed successfully</td>
</tr>
<tr>
<td>404</td>
<td>Not Found: if the transaction doesn't exist</td>
</tr>
<tr>
<td>409</td>
<td>Conflict: Transaction did not commit successfully</td>
</tr>
<tr>
<td>410</td>
<td>Gone: Transaction expired</td>
</tr>
</tbody>
</table>

### DELETE

**Rollback and close an open transaction**

#### Example:

```bash
curl -i -X DELETE "http://localhost:8080/rest/fcr:tx/ce4bb2bf-8ced-4c7d-b281-f2132e3064bb"
```

#### Response:

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>No Content: if the transaction is closed successfully</td>
</tr>
</tbody>
</table>

### Status:

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>No Content: if the transaction is renewed successfully</td>
</tr>
<tr>
<td>404</td>
<td>Not Found: if the transaction doesn't exist</td>
</tr>
<tr>
<td>410</td>
<td>GONE: TRANSACTION EXPIRED</td>
</tr>
<tr>
<td>Status Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>204</td>
<td>No Content: if the transaction is discarded successfully</td>
</tr>
<tr>
<td>410</td>
<td>Gone: if the transaction has already been committed or rolled back</td>
</tr>
<tr>
<td>404</td>
<td>Not Found: if the transaction does not exist</td>
</tr>
</tbody>
</table>