REST API

- Introduction
- API-A Methods
  - describeRepository
  - findObjects
  - getDatastreamDissemination
  - getDissemination
  - getObjectHistory
  - getObjectProfile
  - listDatastreams
  - listMethods
  - resumeFindObjects
- API-M Methods
  - addDatastream
  - addRelationship
  - compareDatastreamChecksum
  - export
  - getDatastream
  - getDatastreamHistory
  - getDatastreams
  - getNextPID
  - getObjectXML
  - getRelationships
  - ingest
  - modifyDatastream
  - modifyObject
  - purgeDatastream
  - purgeObject
  - purgeRelationship
  - setDatastreamState
  - setDatastreamVersionable
  - Validate
- Utility Methods
  - Upload
- WADL

Introduction

The Fedora REST API exposes a subset of the Fedora Access and Management APIs as a RESTful (Representational State Transfer) Web Service. For release 3.2, the Fedora REST API has been upgraded to Beta status. With this change the REST API is no longer optional, it is enabled by default as are all of the other Fedora APIs. The primary reasons behind Beta status are the need for more robust testing, as well as the understanding that in a future release the REST API will likely subsume API-A-Lite and API-M-Lite, thus changing the API somewhat.

For examples of how to use the REST API programmatically, please refer to the TestRESTAPI test class (http://fedora-commons.svn.sourceforge.net/viewvc/*checkout*/fedora-commons/fedora/tags/release-3.0/src/test/junit/fedora/test/api/TestRESTAPI.java).

Ensure DC, RELS-EXT and RELS-INT are versionable if using Managed Content



Due to an outstanding bug FCREPO-849, if you use Managed Content for DC, RELS-EXT or RELS-INT then please make sure these datastreams are versionable (the default setting for versionable is "true", so if you haven't specified this datastream property then you are safe). Ensure that you don't inadvertently set this property to "false" for these datastreams when using the API methods.

2xx Responses only please



The HTTP Response portion of each method description listed below indicates the response on success. Unsuccessful calls will produce non-200 response codes appropriate to the error case. If, however, your client software has difficulty processing non-200 responses (such as is the case with Adobe's Flash Player) adding the query parameter 'flash=true' to any method will ensure that all responses are in the 200 range. In the event of an error, the response code will be set to 200 and the response body will include the error message followed by "::ERROR".

POST Replacement



If the client with which you are working does not support use of the PUT and/or DELETE HTTP methods but does allow you to set headers on the HTTP request, you can use POST replacement to make PUT and DELETE calls. To do this, simply set the X-HTTP-Method-Override request header to the correct method value (PUT or DELETE) and perform a POST request. Your request will be handled by the REST API as if it were a PUT or DELETE.

Removal of .xml shortcut



For release 3.3 the `.xml` shortcut has entirely been removed from the REST API due to functional inconsistencies (see here for more details. If your client uses this shortcut please change it to use the format parameter (?format=xml).

URL-Encoding



The REST API requires that parameters - including path parameters - are URL-encoded. Particularly this is important if you have any PIDs that use escaped-octets in the PID name. In this case the "%" character should be URL-encoded as "%25", eg a PID "changeme:1234%2F56" should be URL-encoded as "changeme:1234%252F56". The ":" PID namespace separator character does not require URL-encoding as it has no special meaning when used in the path component of HTTP URIs; however some software library URL-encoding methods will URL-encode this to %3A - that's not a problem, both ":" and "%3A" will be accepted by the REST API.

API-A Methods

describeRepository

Not implemented

findObjects

URL Syntax

/objects ? [terms | query] [maxResults] [resultFormat] [pid] [label] [state] [ownerId] [cDate] [mDate] [dcmDate] [title] [creator] [subject] [description] [publisher] [contributor] [date] [type] [format] [identifier] [source] [language] [relation] [coverage] [rights]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
terms	a phrase represented as a sequence of characters (including the ? and * wildcards) for the search. If this sequence is found in any of the fields for an object, the object is considered a match. Do NOT use this parameter in combination with the "query" parameter		
query	a sequence of space-separated conditions. A condition consists of a metadata element name followed directly by an operator, followed directly by a value. Valid element names are (pid, label, state, ownerId, cDate, mDate, dcmDate, title, creator, subject, description, publisher, contributor, date, type, format, identifier, source, language, relation, coverage, rights). Valid operators are: contains (), equals (=), greater than (>), less than (<), greater than or equals (>=), less than or equals (<=). The contains () operator may be used in combination with the ? and * wildcards to query for simple string patterns. Space-separators should be encoded in the URL as %20. Operators must be encoded when used in the URL syntax as follows: the (=) operator must be encoded as %3D, the (>) operator as %3E, the (<) operator as %3C, the (>=) operator as %3E%3D, the (<=) operator as %3C%3D, and the (~) operator as %7E. Values may be any string. If the string contains a space, the value should begin and end with a single quote character ('). If all conditions are met for an object, the object is considered a match. Do NOT use this parameter in combination with the "terms" parameter		
maxResults	the maximum number of results that the server should provide at once. If this is unspecified, the server will default to a small value	25	
resultFormat	the preferred output format	html	xml, html
pid	if true, the Fedora persistent identifier (PID) element of matching objects will be included in the response	false	true, false
label	if true, the Fedora object label element of matching objects will be included in the response	false	true, false
state	if true, the Fedora object state element of matching objects will be included in the response	false	true, false
ownerId	if true, each matching objects' owner id will be included in the responsefalsetrue, false	false	true, false
cDate	if true, the Fedora create date element of matching objects will be included in the response	false	true, false
mDate	if true, the Fedora modified date of matching objects will be included in the response	false	true, false
dcmDate	if true, the Dublin Core modified date element(s) of matching objects will be included in the response	false	true, false
title	if true, the Dublin Core title element(s) of matching objects will be included in the response	false	true, false
creator	if true, the Dublin Core creator element(s) of matching objects will be included in the response	false	true, false
subject	if true, the Dublin Core subject element(s) of matching objects will be included in the response	false	true, false
description	if true, the Dublin Core description element(s) of matching objects will be included in the response	false	true, false
publisher	if true, the Dublin Core publisher element(s) of matching objects will be included in the response	false	true, false
contributor	if true, the Dublin Core contributor element(s) of matching objects will be included in the response	false	true, false
date	if true, the Dublin Core date element(s) of matching objects will be included in the response	false	true, false
type	if true, the Dublin Core type element(s) of matching objects will be included in the response	false	true, false
format	if true, the Dublin Core format element(s) of matching objects will be included in the response	false	true, false
identifier	if true, the Dublin Core identifier element(s) of matching objects will be included in the response	false	true, false
source	if true, the Dublin Core source element(s) of matching objects will be included in the response	false	true, false
language	if true, the Dublin Core language element(s) of matching objects will be included in the response	false	true, false
relation	if true, the Dublin Core relation element(s) of matching objects will be included in the response	false	true, false
coverage	if true, the Dublin Core coverage element(s) of matching objects will be included in the response	false	true, false
rights	if true, the Dublin Core rights element(s) of matching objects will be included in the response	false	true, false

Examples

/objects?terms=demo&pid=true&subject=true&label=true&resultFormat=xml

/objects?query=title%7Erome%20creator%7Estaples&pid=true&title=true&creator=true

/objects?query=pid%7E*1&maxResults=50&format=true&pid=true&title=true

getDatastreamDissemination

URL Syntax

/objects/{pid}/datastreams/{dsID}/content ? [asOfDateTime] [download]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
asOfDateTime	indicates that the result should be relative to the digital object as it existed at the given date and time		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ
download	If true, a content-disposition header value "attachment" will be included in the response, prompting the user to save the datastream as a file. A content-disposition header value "inline" will be used otherwise. The filename used in the header is generated by examining in order: RELS-INT for the relationship fedora-model:downloadFilename, the datastream label, and the datastream ID. The file extension (apart from where the filename is specified in RELS-INT) is determined from the MIMETYPE. The order in which these filename sources are searched, and whether or not to generate an extension from the MIMETYPE, is configured in fedora.fcfg. The file used to map between MIMETYPEs and extensions is mime-to-extensions.xml located in the server config directory.		

Examples

/objects/demo:29/datastreams/DC/content

/objects/demo:29/datastreams/DC/content?asOfDateTime=2008-01-01

getDissemination

URL Syntax

/objects/{pid}/methods/{sdefPid}/{method} ? [method parameters]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{sdefPid}	persistent identifier of the sDef defining the methods		
{method}	method to invoke		
method parameters	any parameters required by the method		

Examples

/objects/demo:29/methods/demo:27/resizeImage?width=100

/objects/demo:SmileyEarring/methods/demo:DualResolution/fullSize

getObjectHistory

URL Syntax

/objects/{pid}/versions ? [format]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
format	the preferred output format	html	xml, html

Examples

/objects/demo:29/versions

/objects/demo:29/versions?format=xml

getObjectProfile

URL Syntax

/objects/{pid} ? [format] [asOfDateTime]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
format	the preferred output format	html	xml, html
asOfDateTime	indicates that the result should be relative to the digital object as it existed on the given date		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ

Examples

/objects/demo:29

/objects/demo:29?format=xml

/objects/demo:29?asOfDateTime=2008-01-01

listDatastreams

URL Syntax

/objects/{pid}/datastreams ? [format] [asOfDateTime]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
format	the preferred output format	html	xml, html
asOfDateTime	indicates that the result should be relative to the digital object as it existed on the given date		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ

Examples

/objects/demo:35/datastreams

/objects/demo:35/datastreams?format=xml&asOfDateTime=2008-01-01T05:15:00Z

listMethods

URL Syntax

1. /objects/{pid}/methods ? [format] [asOfDateTime]
2. /objects/{pid}/methods/{sdefPid} ? [format] [asOfDateTime]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{sdefPid}	persistent identifier of the SDef defining the methods		
format	the preferred output format	html	xml, html
asOfDateTime	indicates that the result should be relative to the digital object as it existed on the given date		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ

Examples

/objects/demo:29/methods

/objects/demo:29/methods?format=xml&asOfDateTime=2008-01-01T05:15:00Z

/objects/demo:29/methods/demo:27

/objects/demo:29/methods/demo:27?format=xml&asOfDateTime=2008-01-01T05:15:00Z

resumeFindObjects

URL Syntax

/objects ? [sessionToken] [all findObjects options]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
sessionToken	the identifier of the session to which the search results are being returned		
all findObjects options	all of the same options are available for resumeFindObjects as for findObjects		

Examples

/objects?terms=*&format=xml&pid=true&subject=true&label=true&sessionToken=xyz\\\\

API-M Methods

addDatastream

URL Syntax

/objects/{pid}/datastreams/{dsID} ? [controlGroup] [dsLocation] [altIDs] [dsLabel] [versionable] [dsState] [formatURI] [checksumType] [checksum] [mimeType] [logMessage]

HTTP Method

POST

HTTP Response

201

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
controlGroup	one of "X", "M", "R", or "E" (Inline *X*ML, *M*anaged Content, *R*edirect, or *E*xternal Referenced)	X	X, M, R, E
dsLocation	location of managed or external datastream content		
altIDs	alternate identifiers for the datastream		
dsLabel	the label for the datastream		
versionable	enable versioning of the datastream	true	true, false
dsState	one of "A", "I", "D" (*A*ctive, *I*nactive, *D*eleted)	A	A, I, D
formatURI	the format URI of the datastream		
checksumType	the algorithm used to compute the checksum	DEFAULT	DEFAULT, DISABLED, MD5, SHA-1, SHA-256, SHA-385, SHA-512
checksum	the value of the checksum represented as a hexadecimal string		
mimeType	the MIME type of the content being added, this overrides the Content-Type request header		
logMessage	a message describing the activity being performed		
multipart file as request content	datastream file (for Managed datastreams)		

Examples

POST: /objects/demo:29/datastreams/NEWDS?controlGroup=X&dsLabel=New (with Multipart file)

POST: /objects/demo:29/datastreams/NEWDS?controlGroup=M&dsLocation=http://example:80/newds&dsLabel=New

addRelationship

URL Syntax

/objects/{pid}/relationships/new ? [subject] [predicate] [object] [isLiteral] [datatype]

HTTP Method

POST

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
subject	subject of the relationship. Either a URI for the object or one of its datastreams	URI of this object	
predicate	predicate of the relationship		
object	object of the relationship		
isLiteral	true if the object of the relationship is a literal, false if it is a URI		true, false
datatype	if the object is a literal, the datatype of the literal (optional)		

Examples

POST /objects/demo:29/relationships/new?subject=info%3afedora%2fdemo%3a29%2fDC&predicate=http%3a%2f%2fwww.example.org%2frels%2fname&object=dublin%20core&isLiteral=true

compareDatastreamChecksum

See #getDatastream

export

URL Syntax

/objects/{pid}/export ? [format] [context] [encoding]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
format	the XML format to export	info:fedora/fedora-system:FOXML-1.1	info:fedora/fedora-system:FOXML-1.1, info:fedora/fedora-system:FOXML-1.0, info:fedora/fedora-system:METSFedoraExt-1.1, info:fedora/fedora-system:METSFedoraExt-1.0, info:fedora/fedora-system:ATOM-1.1, info:fedora/fedora-system:ATOMZip-1.1
context	the export context, which determines how datastream URLs and content are represented	public	public, migrate, archive
encoding	the preferred encoding of the exported XML	UTF-8	

Examples

/objects/demo:29/export

/objects/demo:29/export?context=migrate

getDatastream

URL Syntax

/objects/{pid}/datastreams/{dsID} ? [asOfDateTime] [format] [validateChecksum]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
format	the preferred output format	html	xml, html
asOfDateTime	indicates that the result should be relative to the digital object as it existed on the given date		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ
validateChecksum	verifies that the Datastream content has not changed since the checksum was initially computed. If asOfDateTime is null, Fedora will use the most recent version.	false	true, false

Examples

/objects/demo:29/datastreams/DC

/objects/demo:29/datastreams/DC?format=xml

/objects/demo:29/datastreams/DC?format=xml&validateChecksum=true

getDatastreamHistory

URL Syntax

/objects/{pid}/datastreams/{dsid}/history ? [format]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
format	the preferred output format	html	xml, html

Examples

GET: /objects/changeme:1/datastreams/DC/history

GET: /objects/changeme:1/datastreams/DC/history?format=xml

getDatastreams

Not implemented

getNextPID

URL Syntax

/objects/nextPID ? [numPIDs] [namespace] [format]

HTTP Method

POST

HTTP Response

200

Parameters

Name	Description	Default	Options
numPIDs	the number of pids to retrieve	1	
namespace	the namespace of the requested pid(s)	the default namespace of the repository	
format	the preferred output format	html	xml, html

Examples

POST: /objects/nextPID

POST: /objects/nextPID?numPIDs=5&namespace=test&format=xml

getObjectXML

URL Syntax

/objects/{pid}/objectXML

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		

Examples

/objects/demo:29/objectXML

getRelationships

URL Syntax

/objects/{pid}/relationships ? [subject] [predicate] [format]

HTTP Method

GET

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
subject	subject of the relationship(s). Either a URI for the object or one of its datastreams	URI of this object	
predicate	predicate of the relationship(s), if missing returns all predicates		
format	format of the response	rdf/xml	xml (returns rdf/xml), rdf/xml, n-triples, turtle, sparql

Examples

/objects/demo:29/relationships

/objects/demo:29/relationships?subject=info%3afedora%2fdemo%3a29%2fDC

ingest

URL Syntax

/objects/ [{pid}| new] ? [label] [format] [encoding] [namespace] [ownerId] [logMessage] [ignoreMime]

HTTP Method

POST

HTTP Response

201

Request Content

text/xml

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the object to be created	new (see below)	
new	indicator that either a new PID should be created for this object or that the PID to be used is encoded in the XML included as the body of the request		
label	the label of the new object		
format	the XML format of the object to be ingested	info:fedora/fedora-system:FOXML-1.1, info:fedora/fedora-system:FOXML-1.0, info:fedora/fedora-system:METSFedoraExt-1.1, info:fedora/fedora-system:METSFedoraExt-1.0, info:fedora/fedora-system:ATOM-1.1, info:fedora/fedora-system:ATOMZip-1.1	
encoding	the encoding of the XML to be ingested. If this is specified, and given as anything other than UTF-8, you must ensure that the same encoding is declared in the XML. For example, if you specify "ISO-88591" as the encoding, the XML should start with: <?xml version="1.0" encoding="ISO-8859-1"?>	UTF-8	
namespace	the namespace to be used to create a PID for a new empty object; if object XML is included with the request, the namespace parameter is ignored	the default namespace of the repository	
ownerId	the id of the user to be listed at the object owner		
logMessage	a message describing the activity being performed		
ignoreMime	indicates that the request should not be checked to ensure that the content is XML prior to attempting an ingest. This is provided to allow for client applications which do not indicate the correct Content-Type when submitting a request.	false	true, false
XML file as request content	file to be ingested as a new object		

Notes

Executing this request with no request content will result in the creation of a new, empty object (with either the specified PID or a system-assigned PID). The new object will contain only a minimal DC datastream specifying the dc:identifier of the object.

Examples

POST: /objects/new

POST: /objects

POST: /objects/new?namespace=demo

POST: /objects/test:100?label=Test

modifyDatastream

URL Syntax

/objects/{pid}/datastreams/{dsID} ? [dsLocation] [altIDs] [dsLabel] [versionable] [dsState] [formatURI] [checksumType] [checksum] [mimeType] [logMessage] [ignoreContent] [lastModifiedDate]

HTTP Method

PUT

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
dsLocation	location of datastream content		
altIDs	alternate identifiers for the datastream		
dsLabel	the label for the datastream		
versionable	enable versioning of the datastream	the "versionable" property of the existing datastream	true, false
dsState	one of "A", "I", "D" (*A*ctive, *I*nactive, *D*eleted)	A	A, I, D
formatURI	the format URI of the datastream		
checksum Type	the algorithm used to compute the checksum	DEFAULT	DEFAULT, DISABLED, MD5, SHA-1, SHA-256, SHA-385, SHA-512
checksum	the value of the checksum represented as a hexadecimal string		
mimeType	the MIME type of the content being added, this overrides the Content-Type request header		
logMessage	a message describing the activity being performed		
ignoreCon tent	tells the request handler to ignore any content included as part of the request, indicating that you do not intend to update the datasteam content. This is primarily provided to allow the use of client tools which always require content to be included as part of PUT requests.	false	true, false
lastModifie dDate	date/time of the last (known) modification to the datastream, if the actual last modified date is later, a 409 response is returned		
multipart file as request content	file to replace existing datastream (for Managed datastreams)		

Examples

PUT: /objects/demo:35/datastreams/HIGH (with Multipart file)

PUT: /objects/demo:35/datastreams/HIGH?dsLocation=http://example:80/highDS?logMessage=Update

modifyObject

URL Syntax

/objects/{pid} ? [label] [ownerId] [state] [logMessage] [\lastModifiedDate]

HTTP Method

PUT

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
label	the new object label		
ownerId	the id of the user to be listed at the object owner		
state	the new object state - *A*ctive, *I*nactive, or *D*eleted	A	A, I, D
logMessage	a message describing the activity being performed		
lastModifiedDate	date/time of the last (known) modification to the datastream, if the actual last modified date is later, a 409 response is returned		

Examples

PUT: /objects/demo:29?label=Updated

PUT: /objects/demo:29?state=D?logMessage=Deleted

purgeDatastream

URL Syntax

/objects/{pid}/datastreams/{dsID} ? [startDT] [endDT] [logMessage]

HTTP Method

DELETE

HTTP Response

200 with a string array of the date-time stamps of the versions purged

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
startDT	the (inclusive) start date-time stamp of the range. If not specified, this is taken to be the lowest possible value, and thus, the entire version history up to the endDT will be purged		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ
endDT	the (inclusive) ending date-time stamp of the range. If not specified, this is taken to be the greatest possible value, and thus, the entire version history back to the startDT will be purged		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ
logMessage	a message describing the activity being performed		

Examples

DELETE: /objects/demo:35/datastreams/HIGH

purgeObject

URL Syntax

/objects/{pid} ? [logMessage]

HTTP Method

DELETE

HTTP Response

204

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
logMessage	a message describing the activity being performed		

Examples

DELETE: /objects/demo:29

purgeRelationship

URL Syntax

/objects/{pid}/relationships ? [subject] [predicate] [object] [isLiteral] [datatype]

HTTP Method

DELETE

HTTP Response

200

Return body

Text indicating if the relationship was successfully purged: `true` or `false`

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
subject	subject of the relationship. Either a URI for the object or one of its datastreams	URI of this object	
predicate	predicate of the relationship		
object	object of the relationship		
isLiteral	true if the object of the relationship is a literal, false if it is a URI		true, false
datatype	if the object is a literal, the datatype of the literal (optional)		

Examples

DELETE /objects/demo:29/relationships?subject=info%3afedora%2fdemo%3a29%2fDC&predicate=http%3a%2f%2fwww.example.org%2frels%2fname&object=dublin%20core&isLiteral=true

setDatastreamState

URL Syntax

/objects/{pid}/datastreams/{dsID} ? [dsState]

HTTP Method

PUT

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
dsState	one of "A", "I", "D" (*A*ctive, *I*nactive, *D*eleted)	A	A, I, D

Examples

PUT: /objects/demo:35/datastreams/HIGH?dsState=D

setDatastreamVersionable

URL Syntax

/objects/{pid}/datastreams/{dsID} ? [versionable]

HTTP Method

PUT

HTTP Response

200

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
{dsID}	datastream identifier		
versionable	enable versioning of the datastream	true	true, false

Examples

PUT: /objects/demo:35/datastreams/HIGH?versionable=false

Validate

URL Syntax

/objects/{pid}/validate ? [asOfDateTime]

HTTP Method

GET

HTTP Response

200 (OK) if the validation could be carried out (even if the object is not valid)

404 If some object or datastream could not be carried out

401 If the user credentials was insufficient

400 If the parameters are misformed

409 If one of the relevant objects were locked

500 If something else failed on the server

Return Body

XML, adhering to this schema

```
<xs:schema targetNamespace="http://www.fedora.info/definitions/1/0/access/"
           xmlns="http://www.fedora.info/definitions/1/0/access/"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified">
    <xs:element name="validation">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="asOfDateTime"/>
                <xs:element ref="contentModels"/>
                <xs:element ref="problems"/>
                <xs:element ref="datastreamProblems"/>
            </xs:sequence>
            <xs:attribute name="pid" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="valid" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:boolean"/>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="asOfDateTime">
        <xs:simpleType>
            <xs:restriction base="xs:dateTime"/>
        </xs:simpleType>
    </xs:element>
    <xs:element name="contentModels">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="model" minOccurs="0" maxOccurs="unbounded" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="problems">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="problem" minOccurs="0" maxOccurs="unbounded" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="datastreamProblems">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="datastream" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="datastream">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="problem" minOccurs="0" maxOccurs="unbounded" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="datastreamID" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string"/>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Parameters

Name	Description	Default	Options
{pid}	persistent identifier of the digital object		
asOfDateTime	indicates that the result should be relative to the digital object and the repository as it existed at the given date and time		yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ

Examples

GET /objects/validate/demo:29?asOfDateTime=2008-01-01
Returns HTTP 200 with the body

```
<?xml version="1.0" encoding="UTF-8"?>
<validation pid="demo:29" valid="true">
  <asOfDateTime>2008-01-01T00:00:00.000Z</asOfDateTime>
  <contentModels>
    <model>info:fedora/fedora-system:FedoraObject-3.0</model>
  </contentModels>
  <problems>
  </problems>
  <datastreamProblems>
  </datastreamProblems>
</validation>
```

GET /objects/validate/demo:fail
Returns HTTP 200 with the body. Here the error was a mispelled element in the DC datastream, "dc:titel"

```
<?xml version="1.0" encoding="UTF-8"?>
<validation pid="demo:fail" valid="false">
  <asOfDateTime></asOfDateTime>
  <contentModels>
    <model>info:fedora/fedora-system:FedoraObject-3.0</model>
  </contentModels>
  <problems>
  </problems>
  <datastreamProblems>
    <datastream datastreamID="DC">

      <problem>Encountered schema validation error while parsing datastream 'DC' with the schema from content
model 'fedora-system:FedoraObject-3.0'. The error was 'cvc-complex-type.2.4.a: Invalid
content was found starting with element 'dc:titel'. One of '{"http://purl.org/dc/elements/1.1/":title,
"http://purl.org/dc/elements/1.1/":creator, "http://purl.org/dc/elements/1.1/":subject,
"http://purl.org/dc/elements/1.1/":description, "http://purl.org/dc/elements/1.1/":publisher, "http://purl.org
/dc/elements/1.1/":contributor, "http://purl.org/dc/elements/1.1/":date, "http://purl.org
/dc/elements/1.1/":type, "http://purl.org/dc/elements/1.1/":format, "http://purl.org/dc/elements/1.1/":
identifier, "http://purl.org/dc/elements/1.1/":source, "http://purl.org/dc/elements/1.1/":language,
"http://purl.org/dc/elements/1.1/":relation, "http://purl.org/dc/elements/1.1/":coverage, "http://purl.org/dc
/elements/1.1/":rights}' is expected.'</problem>
    </datastream>
  </datastreamProblems>
</validation>
```

Utility Methods

Upload

URL Syntax

/upload

HTTP Method

POST

HTTP Response

202 and a URI for the uploaded file

Parameters

Multipart file as request content

Examples

POST: /upload (with Multipart file)

WADL

When running your own Fedora server, the REST API WADL is available at **/objects/application.wadl**.

Example:

```
http://localhost:8080/fedora/objects/application.wadl
```