Data Avenue REST API specification

Version 0.1.2

Component deployment name: dataavenue
Authored by: SZTAKI

Changelog

2018-07-17: Version 0.1.0
l Initial release with resource collection

2018-08-21: Version 0.1.1
l Minor changes reflecting comments and suggestions: component name, removal of version postfix

2019-10-22: Version 0.1.2
API extended with rename and get (non-acknowledged) transfers, ackowledge transfers.
Resources file/directory made plural files/directories.

Table of contents

REST API

Data transfer and browsing services in CloudiFactoring are provided by a tool called **Data Avenue**, implemented by MTA SZTAKI.

Data Avenue services are available programmatically through a **REST** (Representational State Transfer) API for the platform, from program codes and shell scripts, respectively.

The REST API **endpoint** is accessible at http://host:8080/dataavenue/rest/, where ”host” is the domain name or IP address of the host, where Data Avenue was deployed.

Data Avenue refers to remote files residing on remote storages using **URIs** (Uniform Resource Identifiers). URIs are of the form: protocol://storage-address/path/ or protocol://storage-address/path/file. Directory-type URIs (buckets, containers, directory paths) end with the ‘/’ symbol; URI protocol://storage-address/ refers to the “root” path (e.g. for listing all buckets of the user). The remote file or the directory URI on which an operation is to be performed is specified in the ”x-uri” header field sent to the REST API endpoint. For example, ”x-uri: s3://aws.amazon.com/mybucket/myfile.dat” refers to an object “myfile.dat” in bucket “mybucket” on Amazon S3. Note, that Data Avenue presents uniformly different storages regardless of that commonly called “directories” on the target storage are actually buckets (S3), containers (Swift), or sub-directories (SFTP). Also, Data Avenue refers to individual data units as “files” even if they are SFTP files or binary large objects. In the current implementation of Data Avenue, the protocol prefix can be one of: http://, https://, sftp://, swift://, s3://, gsiftp://, srm://, irods://, or lfn://.

To access Data Avenue REST API each HTTP call is required to pass a valid access key, also known as **ticket** or **token**, in each request’s HTTP header field with name “x-key”. For example, in header ”x-key: 123e4567-e89b-12d3-a456-426655440000” the value “123e…” corresponds to the access key of the user sending request to the Data Avenue service. In the context of Cloudifactoring project, this token is to be replaced with “access token” or “authorization code”, respectively, obtained interacting with the Central User Management component of the platform.

Credentials required to authenticate to the remote storage are passed using the ”x-credentials” header field with value containing a JSON string containing credentials data. The required credentials’s fields and values depend on the given storage type to connect to. Data Avenue uses “UserPass” type credentials for username-password authentication (SFTP) and S3 authentication, where username corresponds to the “access key” and password contains the secret key, for example, ”x-credentials: { Type: UserPass, UserID: accesskey, UserPass: secretkey }”.

Authentication to Swift also requires Keystone version, projectName, etc. fields. The same storage may support more than one authentication method (for example, SFTP might support password authentication and SSH key authentication as well). REST API call GET http://host:8080/dataavenue/rest/authentication/*protocol* serves to get the possible authentication types and credential fields, where protocol is the storage type (e.g., s3) to authenticate to.

Data Avenue works on the following resources:

- directories
- files
- attributes
- transfer

The table below summarizes resources and their related HTTP methods, which are detailed in the following sections.

HTTP method/ Resource	GET	POST	PUT	DELETE
directories	**List** directory	Create/**make directory**	**Rename** (in: json)	**Delete directory**
files	**Download** file	**Upload** a new file (in: octet-stream)	Upload and **overwrite** (in: octet-stream) file **Rename** (in: json)	**Delete file**
attributes	Get file or directory **attributes**	Get attributes of multiple files (**list directory** with **details**)	**Modify** file or directory **attributes**	-
transfers	Transfer(s) **status** (id param or all)	**Start** new transfer	**Acknowledge** transfer status	**Abort** transfer
protocols	Get supported **protocols**	-	-	-
authentication	Get **authentication** types for a given protocol	-	-	-
version	Get Data Avenue **version**	-	-	-

In the **examples**, tool “curl” is used to send HTTP messages. HTTP method is set by -X (--request) switch, header parameters are given using the -H (--header) option.

HTTP requests either return HTTP status code only, or JSON contents in response body. On failure (status codes other than 2xx), the response content type is plain/text (Content-Type) with entity body containing the textual description of the error.

Note: resources file/files or directory/directories can be used interchangably. (Singular is supported for legacy reasons, use plural in new developments.)

Note: header names are case insensitive, e.g. „x-key”, „X-Key” or „X-key” can be used interchangibly.

Directories (directory)

List directory: GET /directory

Lists directory contents. The result is a list of files and subdirectory names (strings) contained by the specified directory (x-uri) in JSON format. Subdirectory names end with a "/" symbol. Note that neither size nor other attributes are returned in this listing, only names, which makes possible to list directories even containing 100 000 subentries. See resource section POST /attributes to list directory contents with attributes details. The directory list can also be the root directory (/), which, for example, in the case of S3 storages, lists buckets of the user.

Request

Parameters:

Name	In	Description
X-URI	header	The remote directory URI to be listed For example: -H 'x-uri: s3://aws.amazon.com/mybucket/'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
ACCEPT	header	Response content type For example: -H 'Accept: application/json'

Response

Name	In	Description
JSON response	body	Directory contents listing in JSON format

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request failed due to invalid remote directory syntax (x-uri) or the lack of permission to list
- 401 - Unauthorized: The access token or credentials for the target storage is invalid
- 404 - Not found: The remote resource (remote directory not be found on the target storage)
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

List bucket contents of “mybucket” on an S3 storage.

Request:
curl -X GET -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,UserID:...}' -H "x-uri: s3://aws.amazon.com/mybucket/" http://host:8080/dataavenue/rest/directory

Response:
["file1", "file2", "file3", "subdir1/", "subdir2/", "subdir3/"]

Create directory: POST /directory

Creates a directory in the remote storage. The x-uri parameter must end with a “/” symbol, i.e. it must be a directory-type URI.

Request

Parameters:

Name	In	Description
X-URI	header	The remote directory URI to be created For example: -H ‘x-uri: s3://aws.amazon.com/mybucket/’
X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’

X-CREDENTIALS	header	Credentials required to authenticate to the remote storage

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done (e.g. target directory already exists)
- 401 - Unauthorized: No access token provided or credentials to the target storage were invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Create bucket with the name of “mybucket” on an S3 storage

Request:
curl -X POST -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,UserID:...}' -H "x-uri: s3://aws.amazon.com/mybucket/" http://host:8080/dataavenue/rest/directory

Response:
HTTP 200 - OK
Bucket created.

Rename directory: PUT/directory

Renames a directory to a new name.

Request

Parameters:

Name	In	Description

X-URI	header	The remote URI of the directory to be renamed For example: -H 'x-uri: s3://aws.amazon.com/mybucket/myfolder'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
CONTENT-TYPE	header	Request content type: -H 'Content-Type: application/json'
FILE CONTENTS	body	JSON, e.g.: {newName: 'myfoldernewname'}

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Rename remote file "myfolder" on an S3 storage to "myfoldernewname".

Request:
curl -X PUT -H 'x-key: ...' -H 'x-credentials:{Type:UserPass, ...}' -H 'x-uri: s3://aws.amazon.com/mybucket/myfile' -H 'Content-Type: application/json' -d '{newName: myfoldernewname}' http://host:8080/dataavenue/rest/directories

Response:
HTTP 200 - OK
Directory renamed.

Delete directory: DELETE/directory

Deletes a directory (recursively) from the target storage. Note that the remote directory, as well as all the files it contains, and all its subdirectories will be removed.

Request

Parameters:

Name	In	Description
X-URI	header	The remote directory URI to be deleted For example: -H 'x-uri: s3://aws.amazon.com/mybucket/'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The remote resource not found on the target storage
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Delete “mybucket” from an S3 storage.

Request:
curl -X DELETE -H 'x-key: ...' -H 'x-credentials:{Type:...}' -H "x-uri: s3://aws.amazon.com/mybucket/" http://host:8080/dataavenue/rest/directory

Response:
HTTP 200 - OK
Bucket deleted.

Files (file)

Download file: GET/file

Download file contents. This request returns remote file contents as a binary stream (application/octet-stream), which allows users to download files (save to local disk) from remote storages.

Request

Parameters:

Name	In	Description
X-URI	header	The remote file URI to be downloaded For example: -H ‘x-uri: s3://aws.amazon.com/mybucket/myfile.dat’

X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
ACCEPT	header	Response content type: -H ‘Accept: application/octet-stream’

Response

Name	In	Description
File contents	body	Remote file contents

Status codes:
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The remote resource not found on the target storage
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Download file “file.dat” from an S3 storage to local file “downloaded”.

Request:

curl -X GET -H 'x-key: ...' -H 'x-credentials:{Type:...}' -H ‘Accept: application/octet-stream’-H "x-uri: s3://aws.amazon.com/mybucket/myfile.dat" -o downloaded http://host:8080/dataavenue/rest/file

Response:
HTTP 200 - OK
Remote file saved as file “downloaded”.

Upload file: POST/file

Upload a (new) file. The remote file must not exist as this operation is expected to create the new resource.

Request

Parameters:

Name	In	Description
X-URI	header	The remote file URI to be uploaded For example: -H ‘x-uri: s3://aws.amazon.com/mybucket/myfile.dat’
X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
CONTENT-TYPE	header	Request content type: -H ‘Content-Type: application/octet-stream’
CONTENT-LENGTH	header	File size in bytes
FILE CONTENTS	body	Binary file contents

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done (e.g., target file already exists)
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example:

Upload a local file “1MB.dat” with remote name “myfile” to an S3 storage.
Request:
curl -X POST -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,UserID:...}' -H "x-uri: s3://aws.amazon.com/mybucket/myfile" -H 'Content-Type: application/octet-stream' --data-binary @1MB.dat http://host:8080/dataavenue/rest/file

Response:
HTTP 200 - OK
Local file “1MB.dat” uploaded with name “myfile” in bucket “mybucket”.

Overwrite file: PUT/file

Uploads (and overwrites) remote file contents. If the remote file already exists, this method will overwrite its contents with the new file contents, otherwise the method will create a new remote file (as it would be done with POST method).

Request

Parameters:

Name	In	Description

X-URI	header	The remote file URI to be uploaded For example: -H 'x-uri: s3://aws.amazon.com/mybucket/myfile.dat'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
CONTENT-TYPE	header	Request content type: -H 'Content-Type: application/octet-stream'
CONTENT-LENGTH	header	File size in bytes
FILE CONTENTS	body	Binary file contents

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Overwrite remote file “myfile” on an S3 storage with contents of local file “1MB.dat”. If the remote file does not exist yet, it will be created.

Request:
curl -X PUT -H 'x-key: ...' -H 'x-credentials:{Type:UserPass, ...}' -H "x-uri: s3://aws.amazon.com/mybucket/myfile" -H 'Content-Type: application/octet-stream' --data-binary @1MB.dat http://host:8080/dataavenue/rest/file

Response:
HTTP 200 - OK
Local file “1MB.dat” uploaded with name “myfile” in bucket “mybucket”.

Rename file: PUT/file

Renames a file to a new name.

Request

Parameters:

Name	In	Description
X-URI	header	The remote file URI to be renamed For example: -H ‘x-uri: s3://aws.amazon.com/mybucket/myfile.dat’
X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
CONTENT-TYPE	header	Request content type: -H ‘Content-Type: application/json’
FILE CONTENTS	body	JSON, e.g.: {newName: ’newFilename’}

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Rename remote file “myfile” on an S3 storage to “myfilenewname”.

Request:
curl -X PUT -H 'x-key: ...' -H 'x-credentials:{Type:UserPass, ...}' -H ’x-uri: s3://aws.amazon.com/mybucket/myfile’ -H 'Content-Type: application/json' -d '{newName: myfilenewname}' http://host:8080/dataavenue/rest/files

Response:
HTTP 200 - OK
File renamed.

Delete file: DELETE/file

Deletes a remote file.

Request

Parameters:

Name	In	Description
X-URI	header	The remote file URI to be deleted For example: -H ‘x-uri: s3://aws.amazon.com/mybucket/myfile.dat’

X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The remote resource not found on the target storage
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Delete remote file “myfile” from an S3 storage.

Request:
curl -X DELETE -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: s3://aws.amazon.com/mybucket/myfile" http://host:8080/dataavenue/rest/file

Response:
HTTP 200 - OK
Remote file “myfile” in bucket “mybucket” deleted.

Attributes

Get attributes: GET/attributes

Gets URI attributes. The URI can be either a directory-type URI (ending with /) or a file URI. Attributes such as size, last modification date, etc. are returned in a JSON map (record).

Note that what attributes are available depends on the actual remote storage; it is not guaranteed that all attributes will be provided for every storage.

Request

Parameters:

Name	In	Description
X-URI	header	The remote file or the directory URI on which an operation is to be performed For example: -H 'x-uri: s3://aws.amazon.com/mybucket/myfile.dat'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
ACCEPT	header	Response content type: -H 'Accept: application/json

Response

Name	In	Description
URI attributes in JSON	body	Remote file or directory attributes
name	json	Name of the file of directory (string)
size	json	File size in bytes (number)
date	json	Creation or last modification time (POSIX epoch in ms, number)
perm	json	Permissions (POSIX permissions string, e.g. drwx---r--; starting with "d" if directory, string)

other	json	Other meta information such as owner, group, etc. (string)

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done (e.g., target file or directory does not exist)
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The remote resource not found on the target storage
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Get attributes of a remote directory (bucket “myucket”).

Request:
curl -X GET -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: s3://aws.amazon.com/mybucket/" http://host:8080/dataavenue/rest/attributes

Response:
HTTP 200 - OK
{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"mybucket/","perm":"drw-------","date":1522067955162,"size":0}

Get attributes of a remote file “myfile” in bucket “mybucket”.

Request:
curl -X GET -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: s3://aws.amazon.com/mybucket/myfile." http://host:8080/dataavenue/rest/attributes

Response:
HTTP 200 - OK
{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"myfile","perm":"-rw-------","date":1528274024418,"size":5}

Modify attributes: PUT/attributes

Modifies file or directory attributes. In the request, a JSON record is to be sent containing only those fields that are to be changed; other fields must be missing (original attributes left unchanged). The URI can be either a remote directory or a file.

Request

Parameters:

Name	In	Description
X-URI	header	The remote file or the directory URI whose attributes is to be changed For example: -H 'x-uri: s3://aws.amazon.com/mybucket/myfile.dat'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
ATTRIBUTES	body	Attributes to be changed and new values in JSON (map)

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The remote resource not found on the target storage
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Change attributes of file "myfile" (in directory "/tmp") to add read-write permissions to everyone on an sftp site .

Request:
curl -X PUT -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: sftp://sftp-host.com/tmp/myfile" -d '{"perm":"-rw----rw-"}' http://host:8080/dataavenue/rest/attributes

Response:
HTTP 200 - OK
Changed file attributes.

List attributes: POST/attributes

Get attributes of multiple files. This request works like directory listing (GET/directory), but the result is now a list of subentries with details (attributes) such as file size and last modification date. The URI must be a directory-type URI, ending with "/".

Request

Parameters:

Name	In	Description
X-URI	header	The remote the directory URI to list For example: -H 'x-uri: s3://aws.amazon.com/mybucket/'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the remote storage
ACCEPT	header	Response content type: -H 'Accept: application/json

FILE LIST	body (optional)	List of file names whose attributes are to return (JSON array of strings). If this list is empty, attributes of all subentries (files and subdirectories) are to return.

Response

Name	In	Description
List of attributes in JSON	body	List of file and subdirectory attributes (list of JSON maps). For attributes see GET/attributes.

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The remote resource not found on the target storage
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Get attributes of all subentries in bucket “testbucket” on an S3 storage.

Request:

curl -X POST -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: s3://s3backup.lpds.sztaki.hu/testbucket/" http://host:8080/dataavenue/rest/attributes

Response:

HTTP 200 - OK

[{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"ciao.txt","perm":"-rw-------","date":1522068061514,"size":5},{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"data-avenue-docker-compose-latest.tar.gz","perm":"-rw-------","date":1528188940169,"size":2958},{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"hello.txt","perm":"-rw-------","date":1528274024418,"size":5}]

Get attributes of files “hello.txt” and ciao.txt in bucket “testbucket” on an S3 storage.

Request:

curl -X POST -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: s3://s3backup.lpds.sztaki.hu/testbucket/" -H 'Content-type: application/json' -d '[hello.txt, ciao.txt]' http://host:8080/dataavenue/rest/attributes

Response:
HTTP 200 - OK
[{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"ciao.txt","perm":"-rw-------","date":1522068061514,"size":5},{"other":"Owner: Akos Hajnal (ahajnal@sztaki.hu)","name":"hello.txt","perm":"-rw-------","date":1528274024418,"size":5}]

Transfers

Transfer status: GET/transfers/{transferId}

Returns transfer status. This method returns transfer details (state, progress, failure) in a JSON record. The identifier of the transfer is returned by POST/transfers.

Request

Parameters:

Name	In	Description
X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
ACCEPT	header	Response content type: -H ‘Accept: application/json
TRANSFER ID	path parameter	The identifier of the transfer whose status is to return

Response

Name	In	Description
Transfer status in JSON	body	Transfer status details

status	json	Transfer status (CREATED|SCHEDULED|DONE|RUNNING|FAILED|ABORTED|RETRY) (string)
started	json	Transfer start time (POSIX epoch in ms, number)
ended	json	Transfer end time if transfer completed/failed (POSIX epoch in ms, number)
serverTime	json	Current server time (POSIX epoch in ms, number)
source	json	Source URI to be transferred (string)
target	json	Target URI to where source is to be transferred (string)
bytesTransferred	json	Bytes transferred successfully so far (number)
size	json	Total size to be transferred in bytes (number)

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The transfer id is invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Get status of transfer with id 6f6f75f7-0e1b-4cfd-bd0f-b681036ec877.

Request:

curl -X GET -H 'X-Key: …' http://host:8080/dataavenue/rest/transfers/6f6f75f7-0e1b-4cfd-bd0f-b681036ec877

Response:

HTTP 200 - OK

{"bytesTransferred":1048576,"source":"s3://storage_a_ip:80/sourcebucket/1MB.dat","status":"DONE","serverTime":1507637326644,"target":"s3://storage_b_ip:80/targetbucket/1MB.dat","ended":1507637273245,"started":1507637271709,"size":1048576}

Non-acknowledged transfers statuses: GET/transfers/

Returns transfer statuses of all not yet acknowledged transfers of the user identified by „x-key”. This method returns the list of transfers' details (state, progress, failure) in a JSON array.

Request

Parameters:

Name	In	Description
X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
ACCEPT	header	Response content type: -H ‘Accept: application/json

Response

Name	In	Description
Transfers status in JSON	body	Transfers status details as JSON list. Each element is a record of the following fields.
status	json	Transfer status (CREATED|SCHEDULED|DONE|RUNNING|FAILED|ABORTED|RETRY) (string)
started	json	Transfer start time (POSIX epoch in ms, number)
ended	json	Transfer end time if transfer completed/failed (POSIX epoch in ms, number)
serverTime	json	Current server time (POSIX epoch in ms, number)
source	json	Source URI to be transferred (string)
target	json	Target URI to where source is to be transferred (string)
bytesTransferred	json	Bytes transferred successfully so far (number)

size	json	Total size to be transferred in bytes (number)

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the target storage are invalid
- 404 - Not found: The transfer id is invalid
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Get statuses of all non-ackowledged transfers.

Request:

curl -X GET -H 'X-Key: …' http://host:8080/dataavenue/rest/transfers/

Response:

HTTP 200 - OK

[{"bytesTransferred":5490304,"size":5490304,"ended":1569935924215,"started":1569935923135,"serverTime":1571740077687,"id":"4eee706d-0961-4156-b6d9-bb813e2f08e5","source":"hdfs://193.224.59.150:9000/user/akos/dataavenue/","target":"s3://192.168.154.2:80/hdfs/","status":"DONE"},{"bytesTransferred":599511,"size":599511,"ended":1569935877011,"started":1569935876591,"serverTime":1571740077687,"id":"073ab625-bddc-439f-a33e-384eb1965408","source":"hdfs://193.224.59.150:9000/user/akos/nagy.pdf","target":"s3://192.168.154.2:80/hdfs/nagy.pdf","status":"DONE"}]

Start transfer: POST/transfers

Start a new transfer. This method is used to transfer data from one storage (source) to another (target). If source URI (given in header x-uri parameter) is a directory then the whole directory will be transferred to target URI (given in request body in JSON), which must also be a directory-type URI. If source is a file, only this file will be transferred to target URI. If target URI is a file, the source will be renamed correspondingly to target URI. If target URI is a directory, file name in the source URI (string from last /) will be kept and used in the target storage as file name. The parameters also allow to move source to target (source will be deleted) or force overwrite target (if exists). On successful request, this method returns the identifier of the transfer just started (string UUID with Content-Type: text/plain). When transferring data within the same storage, Data Avenue tries to use server-side copy (if such service is available, e.g. on S3). If the storage allows third-party transfer (GridFTP), Data

Avenue will use that service (data will be transferred between storages directly).

Request

Parameters:

Name	In	Description
X-URI	header	Source file or the directory URI to transfer For example: -H 'x-uri: s3://aws.amazon.com/mybucket/myfile.dat'
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
X-CREDENTIALS	header	Credentials required to authenticate to the source storage
CONTENT TYPE	header	Request content type: -H 'Content-Type: application/json
TRANSFER TARGET AND OPTIONS	body	Target URI and transfer options in JSON format (map)
TARGET	json	Target file or directory URI (string)
CREDENTIALS	json	Credentials to target storage (JSON map)

OVERWRITE (optional)	json	Overwrite target URI if exists (boolean, default=false)
MOVE (optional)	json	Move source URI to target URI (boolean, default=false)

Response

Name	In	Description
Transfer id	body	Transfer identifier string (UUID)

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided or credentials to the source or target storage are invalid
- 404 - Not found: The source or target URI not found on the source or target storage, respectively
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Transfer a file “1MB.dat” from the bucket “sourceBucket” on storage s3.source.host to bucket “targetBucket” storage on target storage s3.target.host.

Request:

```
curl -X POST -H 'x-key: ...' -H 'x-credentials:{Type:UserPass,...}' -H "x-uri: s3://s3.source.host/sourcebucket/1MB.dat" -H 'Content-type: application/json' --data '{target:”s3://s3.target.host/targetbucket/”,credentials:”{Type:UserPass,...}”,overwrite:false,move:false}' http://host:8080/dataavenue/rest/transfers
```

Response:

```
HTTP 200 - OK
6f6f75f7-0e1b-4cfd-bd0f-b681036ec877
```

Ackowledge transfer state: PUT/transfers/{transferId}

Acknowledge a transfer and don't return its status when non-ackowledged transfers are queried.

Request

Parameters:

Name	In	Description
X-KEY	header	Access key (token) to authenticate to REST API For example: -H 'x-key: 123e4567-e89b-...'
TRANSFER ID	path parameter	The identifier of the transfer to ackowledge

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided
- 404 - Not found: The remote resource (transfer id) not found
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example

Abort a transfer with id 6f6f75f7-0e1b-4cfd-bd0f-b681036ec877.

Request:

curl -X POST -H 'X-Key: …' http://host:8080/dataavenue/rest/transfers/6f6f75f7-0e1b-4cfd-bd0f-b681036ec877

Response:

HTTP 200 - OK

Transfer status acknowledged.

Abort transfer: DELETE/transfers/{transferId}

Aborts an ongoing transfer.

Request

Parameters:

Name	In	Description
X-KEY	header	Access key (token) to authenticate to REST API For example: -H ‘x-key: 123e4567-e89b-...’
TRANSFER ID	path parameter	The identifier of the transfer to abort

Response

Name	In	Description
HTTP status code	status	Request completed or failed

Status codes

Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: The request is invalid or the request cannot be done
- 401 - Unauthorized: No access token provided
- 404 - Not found: The remote resource (transfer id) not found
- 500 - Internal server error: Server-side error (e.g., database not accessible)

Example
Abort a transfer with id 6f6f75f7-0e1b-4cfd-bd0f-b681036ec877.

Request:
curl -X DELETE -H 'X-Key: …' http://host:8080/dataavenue/rest/transfers/6f6f75f7-0e1b-4cfd-bd0f-b681036ec877

Response:
HTTP 200 - OK
Transfer aborted.

Authentication

Version: GET /authentication/{protocol}
Returns authentication types and fields for a given protocol.

Request
Parameters:

Name	In	Description
ACCEPT	header	Response content type: -H 'Accept: application/json
PROTOCOL	path parameter	The protocol of which authentication types and fields are to be returned

Response

Name	In	Description
Authentication types	body	List of authentication types and their fields (JSON array of maps)
displayName	json	Display name for authentication type on a GUI
type	json	Name of authentication type. (Corresponds to field "Type" in x-credentials header.)
fields	json	List of fields belonging to this authentication type

keyName	json	Field name of the authentication field. (This keyName must be used in x-credentials header as key when authenticating to this storage.)
displayName	json	Field name label to be displayed on the GUI for this field.
type	json	Field type: text or password. Password type field value must be masked on the GUI.
defaultValue	json	Default value for field keyName

Status codes
Success:

- 200 - OK: Request was successful

Error:

- 400 - Bad Request: Protocol is unknown or not supported

Example

Get authentication type for protocol S3.

Request:
curl -X GET http://host:8080/dataavenue/rest/authentication/s3

Response:
HTTP 200 - OK
[{"displayName":"S3 authentication","type":"UserPass","fields":[{"keyName":"UserID","type":"text","defaultValue":"","displayName":"Access key"},{"keyName":"UserPass","type":"password","defaultValue":"","displayName":"Secret key"}]}]

Protocols

Version: GET /protocols

Gets list of protocols (in JSON array) that Data Avenue supports.

Request

Parameters:

Name	In	Description
ACCEPT	header	Response content type: -H ‘Accept: application/json

Response

Name	In	Description
List of protocols	body	List of protocols (JSON array of strings)

Status codes
Success:

- 200 - OK: Request was successful

Example

Get supported protocols.

Request:
curl -X GET -H “x-details: true” http://host:8080/dataavenue/rest/protocols

Response:
HTTP 200 - OK
["http","https","sftp","gsiftp","srm","lfn","irods","swift","google","azure","s3"]

Version

Version: GET /version

Gets Data Avenue version and build time.

Request

Parameters:

Name	In	Description
DETAILS (optional)	header	Get server details as well, such as CPU cores, heap size (boolean, default=false)

Response

Name	In	Description
Version response	body	Version and build time string

Status codes
Success:

- 200 - OK: Request was successful

Example

Get Data Avenue version

Request:
curl -X GET -H “x-details: true” http://host:8080/dataavenue/rest/version

Response:
HTTP 200 - OK
3.0.0 (build: 03/07/2018 10:37)

Get Data Avenue version and details

Request:
curl -X GET -H “x-details: true” http://host:8080/dataavenue/rest/v1/version

Response:
HTTP 200 - OK
3.0.0 (build: 03/07/2018 10:37) (CPU cores: 2, heap max: 921MB, heap current: 270MB, heap free: 92MB)
