# GA4GH DRS Client

Oct 09, 2019

# Contents:

The GA4GH DRS Client is a Python-based command-line application for requesting omics data and metadata from web services that are compliant with the Data Repository Service (DRS) API Specification. The DRS API specification, developed by the Global Alliance for Genomics and Health, serves to provide a standardized API framework to allow for interoperability of datasets hosted at different institutions.

Click here for instructions on how to install the client.

CHAPTER 1

# Additional Resources

1. PyPI - The DRS Client is available on the Python Package Index (PyPI)
2. Docker - The DRS Client can be run through a preconfigured image

## 1.1 Installation

This section provides instructions on how to install the DRS command-line client.

As a prerequisite, python 3 and pip must be installed on your system. The application can be installed by running the following from the command line.

1. Install latest distribution from the Python Package Index (PyPI)

```
pip install ga4gh-drs-client
```

2. Confirm installation by executing the drs command

```
drs get
```

The next article explains how to run the drs client.

## 1.2 Usage

The DRS client is executed on the command-line via the following structure:

```
drs get [OPTIONS] URL OBJECT_ID
```

where `[OPTIONS]` represents a set of optional command-line parameters, and `URL` and `OBJECT_ID` represent two position-specific arguments.

### 1.2.1 Arguments and Options

Required command-line arguments:

Table 1: ga4gh-drs-client required arguments

| Parameter | Description |
|---|---|
| URL | Base URL to DRS service (up to but excluding the DRS BasePath ‘/ga4gh/drs/v1’) |
| OBJECT_ID | DRS object identifier |

Optional command-line options:

Table 2: ga4gh-drs-client options

| Parameter | Short Name | Description |
|---|---|---|
| -t | –authtoken | Value of OAuth 2.0 Authorization: Bearer token |
| -d | –download | Flag. If set, download object bytes |
| -x | –expand | Flag. If set, program will recursively traverse inner bundles within the root bundle |
| -l | –logfile | File to which logs should be written |
| -M | –max-threads | Number of concurrent download threads |
| -o | –output-dir | Directory to write downloaded files |
| -m | –output-metdata | File to write object metadata (printed to stdout by default) |
| -S | –silent | Flag. If set, don’t output any messages to console or log file |
| -s | –suppress-ssl-verify | Flag. If set, suppress ssl certification verificiation (NOT RECOMMENDED) |
| -v | –validate-checksum | Flag. If set, perform checksum validation on downloaded objects |
| -V | –verbosity | DEBUG\|INFO\|WARNING\|ERROR Control verbosity of logging |

### 1.2.2 Example Usage

1. Basic Usage, get DRS object and print metadata to screen

```
drs get https://exampledrs.com/ a02568e6-11f8-4493-9880-f51823df09b8
```

2. Write metadata to an output file

```
drs get -m metadata.json https://exampledrs.com/ a02568e6-11f8-4493-9880-f51823df09b8
```

3. Download object bytes, writing output files to the “output” directory

```
drs get -d -o output https://exampledrs.com/ a02568e6-11f8-4493-9880-f51823df09b8
```

4. Use an auth token to access DRS object data/metadata

```
drs get -d -o output -t P8vNFYh6jC https://exampledrs.com/ a02568e6-11f8-4493-9880-
↪f51823df09b8
```

5. Write debug, info, warning, and error logs to a log file

```
drs get -l logfile.txt -V DEBUG https://exampledrs.com/ a02568e6-11f8-4493-9880-
↪f51823df09b8
```

## 1.3 Supported Schemes

According to the DRS Specification, object bytes can be downloaded by multiple access method types. The DRS client supports byte download by different types, indicated by the *type* parameter of *AccessMethod* objects in a *DRSObject's access_methods* array. These access method types correspond to URI schemes. For each *DRSObject*, the client will attempt to download object bytes by each supported scheme in sequence, until the file has been successfully downloaded, or until all download options have been exhausted without success.

Currently, the DRS client supports download by **2** URI schemes/access method types:

Table 3: ga4gh-drs-client supported schemes

| Scheme | Description |
|---|---|
| gs | Google Cloud Storage |
| https | Hypertext Transfer Protocol Secure |

## 1.4 Report and Output

At a high level, the DRS client generates 3 different types of data when executed:

1. Requested object metadata
2. Downloaded files
3. Download status report

### 1.4.1 Requested object metadata

Metadata for the requested DRS object is downloaded as JSON. By default, metadata is printed to screen. If the `-m FILENAME` option is used on the command-line, output will be written to the specified file.

### 1.4.2 Downloaded files

If the `-d` flag is used on the command-line, the client will attempt to download bytes for the DRS object. If the requested object id was a bundle, it will download bytes for all objects in the bundle.

By default, downloaded files are written to the current working directory. If the `-o DIRECTORY` option is used on the command-line, downloaded files are written to the user-specified output directory.

### 1.4.3 Download status report

If the client has attempted to download bytes for one or more DRS objects, a download status report will be written to the output directory. This text file includes a table, one row per downloaded file. Each row indicates whether the file was successfully downloaded, and whether the file passed checksum validation (if validation was requested).

The columns of the download status report are as follows:

Table 4: Download Status Report Columns

| Column # | Field Name | Description |
|---|---|---|
| 1 | ID | ID of DRS object corresponding to downloaded file |
| 2 | Name | Name of DRS object corresponding to downloaded file |
| 3 | Output File | Local file where downloaded bytes were written |
| 4 | Download Status | COMPLETED/FAILED. Indicates whether file was successfully downloaded |
| 5 | Checksum Status | PASSED/FAILED. Indicates whether downloaded file passed checksum validation (if requested) |
| 6 | Hash Algorithm | The hash algorithm used to perform checksum validation |
| 7 | Expected | Digest value according to the DRS service/object metadata |
| 8 | Observed | Digest value computed locally on the downloaded file |

## 1.5 Example Scripts

This page provides links to some example `drs get` commands that will download DRS object metadata and bytes from different DRS services.

NOTES:

- you will need the appropriate auth tokens to successfully run the sample commands
- each script expects an environment variable, `AUTHTOKEN`, the value of which is the OAuth 2.0 authorization token for the DRS service

### 1.5.1 Scripts

1. Download object from Terra Data Repository
2. Download bundle from Terra Data Repository

CHAPTER 2

# Indices and tables

- genindex
- modindex
- search