Package 'dwapi'

August 19, 2018

Title A Client for 'data.world' REST API

Version 0.1.3.1

Description
A set of wrapper functions for 'data.world' REST API endpoints <https://apidocs.data.world>.

Depends R (>= 3.3.0)

Imports httr, readr, rjson, xml2, jsonlite

Suggests knitr, rmarkdown, testthat (>= 2.0.0), mockery, curl

License Apache License 2.0

Encoding UTF-8

LazyData true

URL https://github.com/datadotworld/dwapi-r

BugReports https://github.com/datadotworld/dwapi-r/issues

RoxygenNote 6.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Rafael Pereira [aut, cre],
Triet Le [aut],
Bryon Jacob [aut],
Scott Came [aut]

Maintainer ORPHANED

Repository CRAN

Date/Publication 2018-08-19 08:51:26 UTC

X-CRAN-Original-Maintainer Rafael Pereira <rafael.pereira@data.world>

X-CRAN-Comment Orphaned and corrected on 2018-08-19 as check problems were not corrected despite reminders.

R topics documented:

add_file . . . 4
add_files_by_source . . . 5
add_file_by_source . . . 6
append_data_frame_to_stream . . . 7
append_record_to_stream . . . 8
append_values_to_stream . . . 9
auth_token . . . 10
check_file_summary_response . . . 10
check_project_summary_response . . . 11
check_user_info_response . . . 11
configure . . . 12
create_dataset . . . 12
create_dataset_response . . . 13
create_insight . . . 14
create_project . . . 14
create_project_response . . . 15
dataset_create_request . . . 16
dataset_replace_request . . . 17
dataset_summary_response . . . 18
dataset_update_request . . . 18
delete_dataset . . . 19
delete_file . . . 20
delete_files . . . 20
delete_insight . . . 21
delete_project . . . 22
download_datapackage . . . 22
download_file . . . 23
download_file_as_data_frame . . . 24
download_table_as_data_frame . . . 24
dwapi . . . 25
error_message . . . 27
extract_dataset_key . . . 28
extract_project_key . . . 28
file_batch_update_request . . . 29
file_create_or_update_request . . . 29
file_create_request . . . 30
file_source_create_or_update_request . . . 31
file_source_create_request . . . 31
file_source_summary_response . . . 32
file_summary_response . . . 32
get_dataset . . . 33
get_datasets_user_contributing . . . 33
get_datasets_user_liked . . . 34
get_datasets_user_own . . . 35
get_insight . . . 35
get_insights . . . 36

get_project . . . 37
get_projects_user_contributing . . . 37
get_projects_user_liked . . . 38
get_projects_user_own . . . 39
get_table_schema . . . 40
get_user . . . 40
insight_create_request . . . 41
insight_replace_request . . . 42
insight_summary_response . . . 43
insight_update_request . . . 43
linked_dataset_create_or_update_request . . . 44
link_dataset . . . 45
list_tables . . . 46
parse_success_or_error . . . 46
project_create_request . . . 47
project_replace_request . . . 48
project_summary_response . . . 49
project_update_request . . . 49
replace_dataset . . . 50
replace_insight . . . 51
replace_project . . . 52
sdk_version . . . 53
sparql . . . 53
sql . . . 54
success_message . . . 55
sync . . . 55
table_schema_field_response . . . 56
table_schema_field_update_request . . . 56
table_schema_response . . . 57
table_schema_update_request . . . 57
unlink_dataset . . . 58
update_dataset . . . 59
update_insight . . . 59
update_project . . . 60
update_table_schema . . . 61
upload_data_frame . . . 62
upload_file . . . 62
upload_files . . . 63
user_agent . . . 64
user_info_response . . . 64

Index **65**

add_file *Add a file to a request object.*

Description

Add a file to a request object.

Usage

```
add_file(request, name, url, description = NULL, labels = NULL)

## Default S3 method:
add_file(request, name, url, description = NULL,
  labels = NULL)

## S3 method for class 'file_batch_update_request'
add_file(request, name, url,
  description = NULL, labels = NULL)

## S3 method for class 'dataset_create_request'
add_file(request, name, url,
  description = NULL, labels = NULL)

## S3 method for class 'dataset_replace_request'
add_file(request, name, url,
  description = NULL, labels = NULL)

## S3 method for class 'dataset_update_request'
add_file(request, name, url = NULL,
  description = NULL, labels = NULL)
```

Arguments

request	Request object and container for files.
name	Filename including the file extension. If a file by that name already exists in the dataset, the file will be updated/overwritten.
url	Source URL for file. Optional, if updating existing files.
description	(optional) File description.
labels	(optional) List of file labels ("raw data", "documentation", "visualization", "clean data", "script" or "report").

Value

Modified request object.

Methods (by class)

- `default`: Default implementation
- `file_batch_update_request`: Add a file to a file_batch_update_request objects
- `dataset_create_request`: Add a file to a dataset_create_request object
- `dataset_replace_request`: Add a file to a dataset_replace_request object
- `dataset_update_request`: Add a file to a dataset_update_request object

Examples

```
file_batch_update_req <- dwapi::file_batch_update_request()

file_batch_update_req <- dwapi::add_file(request = file_batch_update_req,
  name = 'file.csv', url = 'https://data.world/file3.csv')

dataset_create_req <- dwapi::dataset_create_request(title='coffeeCounty',
  visibility = 'OPEN', description = 'coffee county , AL - census income' ,
  tags = c('rsdk', 'sdk', 'arr') , license_string = 'Public Domain')

dataset_create_req <- dwapi::add_file(request = dataset_create_req,
  name = 'file4.csv', url = 'https://data.world/file4.csv')

dataset_replace_req <- dwapi::dataset_replace_request(visibility = 'OPEN',
  description = 'updated description', title = 'updated title', files = list())

dataset_replace_req <- dwapi::add_file(request = dataset_replace_req,
  name = 'file4.csv', url = 'https://data.world/file4.csv',
  description = "My 4th csv", labels = list("clean data"))
```

---

add_files_by_source *Add one or more files to a dataset.*

Description

Add one or more files to a dataset.

Usage

```
add_files_by_source(dataset, file_batch_update_req)
```

Arguments

`dataset`	Dataset URL or path.
`file_batch_update_req`	Object of type `file_batch_update_request`.

Value

Object of type `success_message`.

Examples

```
request <- dwapi::file_batch_update_request()
request <- dwapi::add_file(request = request, name = 'file.csv',
     url = 'https://data.world/some_file.csv')
## Not run:
  dwapi::add_files_by_source(dataset = 'user/dataset',
     file_batch_update_req = request)

## End(Not run)
```

`add_file_by_source` *Add a single file to a dataset.*

Description

Add a single file to a dataset.

Usage

```
add_file_by_source(dataset, name, url)
```

Arguments

`dataset`	Dataset URL or path.
`name`	File name including the file extension. If a file by that name already exists in the dataset, the file will be updated/overwritten.
`url`	Source URL of file.

Value

Object of type `success_message`.

Examples

```
## Not run:
  dwapi::add_file_by_source(dataset = 'user/dataset',
    name = 'file.csv', url = 'https://data.world/some_file.csv')

## End(Not run)
```

append_data_frame_to_stream *Append an R data frame to a data.world stream.*

Description

Append an R data frame to a data.world stream. If the data.world API returns an HTTP status of 429 (Too Many Requests), this function uses RETRY to retry the request.

Usage

```
append_data_frame_to_stream(owner_id, dataset_id, stream_id, data_frame,
  retry_times = 3, retry_quiet = FALSE)
```

Arguments

owner_id	User name and unique identifier of the creator of a dataset or project
dataset_id	Dataset unique identifier
stream_id	Stream unique identifier as defined by the user the first time the stream was used. Only lower case letters, numbers and dashes are allowed.
data_frame	The data frame containing the rows to append to the stream
retry_times	the number of times to retry the request
retry_quiet	whether to suppress diagnostic messages during retries

Value

Server response message.

Examples

```
## Not run:
  aDf <- data.frame(ID=1:2, Value=c('One', 'Two'), stringsAsFactors = FALSE)
  dwapi::append_data_frame_to_stream(owner_id = 'user',
    dataset_id = 'dataset', stream_id = 'mystream',
    aDf)
  aDf <- data.frame(ID=1:2, Value=c('One', 'Two'), stringsAsFactors = FALSE)
  dwapi::append_data_frame_to_stream(owner_id = 'user',
    dataset_id = 'dataset', stream_id = 'mystream',
    aDf, retry_times = 10, retry_quiet = TRUE)

## End(Not run)
```

append_record_to_stream *Append a record, consisting of a list of depth one, to a stream.*

Description

Append a record, consisting of a list of depth one, to a stream. The list must consist of atomic vectors of length one. If the data.world API returns an HTTP status of 429 (Too Many Requests), this function uses RETRY to retry the request.

Usage

```
append_record_to_stream(owner_id, dataset_id, stream_id, record,
  retry_times = 3, retry_quiet = FALSE)
```

Arguments

owner_id	User name and unique identifier of the creator of a dataset or project
dataset_id	Dataset unique identifier
stream_id	Stream unique identifier as defined by the user the first time the stream was used. Only lower case letters, numbers and dashes are allowed.
record	the record list
retry_times	the number of times to retry the request
retry_quiet	whether to suppress diagnostic messages during retries

Value

Server response message.

Examples

```
## Not run:
  record <- list(ID=1, Value='One')
  dwapi::append_record_to_stream(owner_id = 'user',
    dataset_id = 'dataset', stream_id = 'mystream',
    record)

## End(Not run)
```

append_values_to_stream *Append a record, consisting of named parameters, to a stream.*

Description

Append a record, consisting of named parameters, to a stream. Each value must be an atomic vector of length one. If the data.world API returns an HTTP status of 429 (Too Many Requests), this function uses RETRY to retry the request.

Usage

```
append_values_to_stream(owner_id, dataset_id, stream_id, retry_times = 3,
  retry_quiet = FALSE, ...)
```

Arguments

owner_id	User name and unique identifier of the creator of a dataset or project
dataset_id	Dataset unique identifier
stream_id	Stream unique identifier as defined by the user the first time the stream was used. Only lower case letters, numbers and dashes are allowed.
retry_times	the number of times to retry the request
retry_quiet	whether to suppress diagnostic messages during retries
...	named parameters giving the variables and values in the record to be streamed

Value

Server response message.

Examples

```
## Not run:
  dwapi::append_values_to_stream(owner_id = 'user',
    dataset_id = 'dataset', stream_id = 'mystream',
    ID=1, Value='One')

## End(Not run)
```

auth_token *Return the currently configured API authentication token*

Description

Used by internal functions to obtain auth token required to make API requests. Will fail with an user error to educate users on how to configure their token.

Usage

```
auth_token()
```

Value

API token

check_file_summary_response *Validate* get_dataset *response files.*

Description

Validate get_dataset response files.

Usage

```
check_file_summary_response(object)
```

Arguments

object	Object of type file_summary_response

Value

TRUE/FALSE depending on validity of object.

check_project_summary_response

Validate `get_project` *response object.*

Description

Validate `get_project` response object.

Usage

```
check_project_summary_response(object)
```

Arguments

`object`	Object of type `project_summary_response`.

check_user_info_response

Validate `get_user` *response object, returning the object if valid, and stopping with an error message if invalid.*

Description

Validate `get_user` response object, returning the object if valid, and stopping with an error message if invalid.

Usage

```
check_user_info_response(object)
```

Arguments

`object`	Object of type `user_info_response`.

Value

the object

configure *Library configuration tool.*

Description

Configuration is not persistent and must be performed for every new R session.

Usage

```
configure(auth_token = NULL)
```

Arguments

auth_token	data.world's API authentication token.

DO NOT SHARE YOUR AUTHENTICATION TOKEN

For your security, do not include your API authentication token in code that is intended to be shared with others.

Call this function via console, always when possible.

If you must call it in code do not include the actual API token. Instead, pass the token via a variable in .Renviron, and do not share your .Renviron file. For example:

```
dwapi::configure(auth_token = Sys.getenv("DW_AUTH_TOKEN"))
```

Examples

```
dwapi::configure(auth_token = "YOUR_API_TOKEN_HERE")
```

create_dataset *Create a new dataset.*

Description

Create a new dataset.

Usage

```
create_dataset(owner_id, create_dataset_req)
```

Arguments

owner_id	data.world user name of the dataset owner.
create_dataset_req	Request object of type dataset_create_request.

Value

Object of type create_dataset_response.

Examples

```
request <- dwapi::dataset_create_request(
  title='testdataset', visibility = 'OPEN',
  description = 'Test Dataset by R-SDK', tags = c('rsdk', 'sdk', 'arr'),
  license_string = 'Public Domain')

request <- dwapi::add_file(request = request, name = 'file4.csv',
  url = 'https://data.world/file4.csv')

## Not run:
dwapi::create_dataset(create_dataset_req = request,
  owner_id = 'user')

## End(Not run)
```

---

create_dataset_response

De-serialize a structured list into create_dataset_reponse object.

Description

De-serialize a structured list into create_dataset_reponse object.

Usage

```
create_dataset_response(structure)
```

Arguments

structure	httr response object.

Value

Object of type create_dataset_response.

create_insight *Create an insight.*

Description

Create an insight.

Usage

```
create_insight(project_owner, project_id, create_insight_req)
```

Arguments

project_owner	ID of project owner
project_id	ID of project to which insight is to be added
create_insight_req	Request object of type insight_create_request.

Value

Object of type create_insight_response.

Examples

```
## Not run:
  dwapi::create_insight(
    project_owner = 'user',
    project_id = 'project',
    insight_create_request(title='My Insight', image_url='https://...'))

## End(Not run)
```

create_project *Create a new project.*

Description

Create a new project.

Usage

```
create_project(owner_id, create_project_req)
```

Arguments

owner_id	data.world user name of the project owner.
create_project_req	Request object of type project_create_request.

Value

Object of type create_project_response.

Examples

```
request <- dwapi::project_create_request(
  title='testproject', visibility = 'OPEN',
  objective = 'Test project by R-SDK', tags = c('rsdk', 'sdk', 'arr'),
  license = 'Public Domain')

request <- dwapi::add_file(request = request, name = 'file4.csv',
  url = 'https://data.world/file4.csv')

## Not run:
dwapi::create_project(create_project_req = request,
  owner_id = 'user')

## End(Not run)
```

---

create_project_response

De-serialize a structured list into create_project_reponse object.

Description

De-serialize a structured list into create_project_reponse object.

Usage

```
create_project_response(structure)
```

Arguments

structure	httr response object.

Value

Object of type create_project_response.

dataset_create_request *Create request object for new datasets.*

Description

Create request object for new datasets.

Usage

```
dataset_create_request(title, visibility, description = "", summary = "",
  tags = list(), license_string = "", file_create_requests = list())
```

Arguments

title	Dataset title (3 to 60 characters).
visibility	Dataset visibility ("PRIVATE" or "OPEN").
description	(optional) Dataset description.
summary	(optional) Dataset summary (markdown supported).
tags	(optional) List of dataset tags (letters, numbers and spaces).
license_string	Dataset license ("Public Domain", "PDDL", "CC-0", "CC-BY", "ODC-BY", "CC-BY-SA", "ODC-ODbL", "CC BY-NC-SA" or Other).
file_create_requests	(optional) List of file_create_request objects.

Value

Request object of type dataset_create_request.

See Also

create_dataset, add_file

Examples

```
request <- dwapi::dataset_create_request(title='datasetid', visibility = 'OPEN',
  description = 'description', tags = c('sdk') , license_string = 'Public Domain')
request <- dwapi::add_file(request = request, name = 'file.csv',
  url = 'http://data.world/file.csv')
```

dataset_replace_request *Create request object for replacing existing datasets.*

Description

Create request object for replacing existing datasets.

Usage

```
dataset_replace_request(title, description = NULL, summary = NULL,
  tags = NULL, license_string = NULL, visibility, files = NULL)
```

Arguments

title	Dataset title
description	(optional) Dataset description.
summary	(optional) Dataset summary (markdown supported).
tags	(optional) List of dataset tags (letters, numbers and spaces).
license_string	Dataset license ("Public Domain", "PDDL", "CC-0", "CC-BY", "ODC-BY", "CC-BY-SA", "ODC-ODbL", "CC BY-NC-SA" or Other).
visibility	Dataset visibility ("PRIVATE" or "OPEN").
files	(optional) List of file_create_request objects.

Value

Request object of type dataset_replace_request.

See Also

replace_dataset, add_file

Examples

```
dataset_put_req <- dwapi::dataset_replace_request(visibility = 'OPEN',
  description = 'updated description', title = 'updated title')
```

`dataset_summary_response` *Deserialize* `get_dataset` *response object.*

Description

Deserialize `get_dataset` response object.

Usage

```
dataset_summary_response(structure)
```

Arguments

`structure`	httr response object.

Value

Object of type `dataset_summary_response`.

See Also

`get_dataset`

`dataset_update_request` *Create request object for updating existing datasets.*

Description

Create request object for updating existing datasets.

Usage

```
dataset_update_request(description = NULL, summary = NULL, tags = NULL,
  license_string = NULL, visibility = NULL, files = NULL)
```

Arguments

`description`	(optional) Dataset description.
`summary`	(optional) Dataset summary (markdown supported).
`tags`	(optional) List of dataset tags (letters, numbers and spaces).
`license_string`	(optional) Dataset license ("Public Domain", "PDDL", "CC-0", "CC-BY", "ODC-BY", "CC-BY-SA", "ODC-ODbL", "CC BY-NC-SA" or Other).
`visibility`	(optional) Dataset visibility ("PRIVATE" or "OPEN").
`files`	(optional) List of `file_create_or_update_request` objects.

Value

Request object of type `dataset_update_request`.

See Also

`update_dataset`, `add_file`

Examples

```
request <- dwapi::dataset_update_request(description = 'description', summary = 'summary',
  tags = list('sdk'), license_string = 'Public Domain')
```

`delete_dataset` *Delete a dataset*

Description

Delete a dataset

Usage

```
delete_dataset(dataset)
```

Arguments

`dataset`	Dataset URL or path.

Value

Object of type `success_message`.

Examples

```
## Not run:
  dwapi::delete_dataset(
    dataset = 'user/dataset')

## End(Not run)
```

delete_file *Delete a single file from a dataset.*

Description

Delete a single file from a dataset.

Usage

```
delete_file(dataset, file_name)
```

Arguments

dataset	Dataset URL or path.
file_name	File name, including the file extension.

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::delete_file(dataset = 'user/dataset',
    file_name = 'file.csv')

## End(Not run)
```

delete_files *Delete one or more files from a dataset.*

Description

Delete one or more files from a dataset.

Usage

```
delete_files(dataset, file_names)
```

Arguments

dataset	Dataset URL or path.
file_names	List of file names, including file extensions.

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::delete_files(dataset = 'user/dataset',
    file_names = list('file1.csv', 'file2.csv'))

## End(Not run)
```

`delete_insight` *Delete an insight.*

Description

Delete an insight.

Usage

```
delete_insight(project_owner, project_id, id)
```

Arguments

`project_owner`	ID of project owner
`project_id`	ID of project of which insight is a component
`id`	ID of insight to be replaced

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::delete_insight(
    project_owner = 'user',
    project_id = 'project',
    id = 'insight')

## End(Not run)
```

delete_project *Delete a project.*

Description

Delete a project.

Usage

```
delete_project(project_owner, project_id)
```

Arguments

project_owner	ID of project owner
project_id	ID of project to delete

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::delete_project(
    project_owner = 'user',
    project_id = 'project')

## End(Not run)
```

download_datapackage *Download data package.*

Description

A data package is a portable dataset file format. Learn more about the Data Package specification at http://frictionlessdata.io/data-packages/.

Usage

```
download_datapackage(dataset, output_path = NULL)
```

Arguments

dataset	Dataset URL or path.
output_path	Directory path, where data package will be saved.

Examples

```
## Not run:
  dwapi::download_datapackage(
    dataset = "user/dataset",
    output_path = tempdir())

## End(Not run)
```

`download_file` *Download file from dataset onto the local file system.*

Description

Download file from dataset onto the local file system.

Usage

```
download_file(dataset, file_name, output)
```

Arguments

`dataset`	Dataset URL or path.
`file_name`	File name, including file extension.
`output`	Local file path, where dataset file will be saved.

Value

Server response message.

Examples

```
## Not run:
  dwapi::download_file(dataset = 'user/dataset',
   file_name = 'file.csv', output = tempfile(fileext = 'csv'))

## End(Not run)
```

download_file_as_data_frame

Download dataset file onto a data frame.

Description

Download dataset file onto a data frame.

Usage

```
download_file_as_data_frame(dataset, file_name)
```

Arguments

dataset	Dataset URL or path.
file_name	File name, including file extension.

Value

Data frame with the contents of CSV file.

Examples

```
## Not run:
  my_df <- dwapi::download_file_as_data_frame(
    dataset = 'user/dataset',
    file_name = 'file.csv')

## End(Not run)
```

download_table_as_data_frame

Download a dataset table onto a data frame.

Description

Download a dataset table onto a data frame.

Usage

```
download_table_as_data_frame(dataset, table_name)
```

Arguments

dataset	Dataset URL or path.
table_name	Table name.

Value

Data frame with data from table.

See Also

list_tables

Examples

```
## Not run:
  table_df <- dwapi::download_table_as_data_frame("user/dataset", "table")

## End(Not run)
```

---

dwapi *dwapi: A client for data.world's REST API.*

Description

The dwapi package makes it easy to access and work with data.world's REST API.

Configuration

The package can be configured with the configure function.

Make sure to configure authentication at the begining of every new R session.

API authentication tokens can be obtained at https://data.world/settings/advanced

API functions

Managing datasets:

1. get_dataset
2. create_dataset
3. update_dataset
4. replace_dataset
5. delete_dataset
6. sync
7. download_datapackage

Managing files:

1. download_file
2. download_file_as_data_frame
3. upload_file
4. upload_data_frame

5. add_files_by_source
6. delete_file

Managing projects:

1. get_project
2. get_projects_user_contributing
3. get_projects_user_liked
4. get_projects_user_own
5. create_project
6. update_project
7. replace_project
8. delete_project

Managing insights:

1. get_insight
2. get_insights
3. create_insight
4. update_insight
5. replace_insight
6. delete_insight

Appending data to streams:

1. append_data_frame_to_stream
2. append_record_to_stream
3. append_values_to_stream

Managing tables and schemas (data dictionary):

1. list_tables
2. get_table_schema
3. update_table_schema
4. download_table_as_data_frame

Querying:

1. sql
2. sparql

Request object constructors

Complex requests are facilitated by request object constructors. The main ones are:

1. dataset_create_request
2. dataset_update_request
3. dataset_replace_request
4. file_batch_update_request
5. file_create_request
6. file_create_or_update_request
7. file_source_create_request
8. file_source_create_or_update_request
9. table_schema_update_request
10. project_create_request
11. project_update_request
12. project_replace_request
13. insight_create_request
14. insight_update_request
15. insight_replace_request
16. linked_dataset_create_or_update_request

---

error_message *Deserialize error response object.*

Description

Deserialize error response object.

Usage

```
error_message(structure)
```

Arguments

structure	httr response object

Value

Object of type error_message

extract_dataset_key *Extract the dataset key from URL or as provided*

Description

Extract the dataset key from URL or as provided

Usage

```
extract_dataset_key(tentative_key)
```

Arguments

tentative_key	key or URL

Value

dataset key extracted form URL or as provided

extract_project_key *Extract the project key from URL or as provided*

Description

Extract the project key from URL or as provided

Usage

```
extract_project_key(tentative_key)
```

Arguments

tentative_key	key or URL

Value

project key extracted form URL or as provided

file_batch_update_request

Create request object for adding or updating dataset files.

Description

Create request object for adding or updating dataset files.

Usage

```
file_batch_update_request(file_sources = list())
```

Arguments

file_sources	(optional) List of file_create_or_update_request objects.

Value

Object of type file_batch_update_request.

See Also

add_files_by_source, add_file

Examples

```
request <- dwapi::file_batch_update_request()
request <- dwapi::add_file(
  request = request, name = 'file.csv', url = 'http://data.world/file.csv')
```

file_create_or_update_request

Create object for adding/updating a dataset file.

Description

Create object for adding/updating a dataset file.

Usage

```
file_create_or_update_request(file_name, url = NULL, description = NULL,
  labels = NULL)
```

Arguments

file_name	File name including the file extension. If a file by that name already exists in the dataset, the file will be updated/overwritten.
url	(optional) Source URL for file.
description	(optional) File description.
labels	(optional) List of file labels ("raw data", "documentation", "visualization", "clean data", "script" or "report").

Value

Object of type file_create_or_update_request.

See Also

file_batch_update_request, update_dataset

Examples

```
file_create_or_update_req <- dwapi::file_create_or_update_request(
  file_name = "file.csv",
  url = "https://data.world/file.csv",
  description = "My updated CSV description",
  labels = list("raw data", "clean data"))
```

---

file_create_request *Create object for adding a dataset file.*

Description

Create object for adding a dataset file.

Usage

```
file_create_request(file_name, url, description = NULL, labels = NULL)
```

Arguments

file_name	File name including the file extension. If a file by that name already exists in the dataset, the file will be updated/overwritten.
url	Source URL for file.
description	(optional) File description.
labels	(optional) List of file labels ("raw data", "documentation", "visualization", "clean data", "script" or "report").

Value

Object of type file_create_request.

See Also

create_dataset, replace_dataset

Examples

```
file_create_req <- dwapi::file_create_request(file_name = "file.csv",
  url = "https://data.world/file.csv",
  description = "My CSV file",
  labels = list("raw data"))
```

---

file_source_create_or_update_request

Create object for adding/updating dataset file sources.

Description

Create object for adding/updating dataset file sources.

Usage

```
file_source_create_or_update_request(url)
```

Arguments

url	Source URL of file.

Value

Object of type file_source_create_or_update_request.

---

file_source_create_request

Create object for adding dataset file sources.

Description

Create object for adding dataset file sources.

Usage

```
file_source_create_request(url)
```

Arguments

url	Source URL of file.

Value

Object of type `file_source_create_request`.

`file_source_summary_response` *Deserialize* `get_dataset` *response file sources.*

Description

Deserialize `get_dataset` response file sources.

Usage

```
file_source_summary_response(structure)
```

Arguments

`structure`	httr response object.

Value

Object of type `file_source_summary_response`

`file_summary_response` *Deserialize* `get_dataset` *response files.*

Description

Deserialize `get_dataset` response files.

Usage

```
file_summary_response(structure)
```

Arguments

`structure`	httr response object.

Value

Object of type `file_summary_response`

get_dataset *Retrieve dataset information.*

Description

Retrieve dataset information.

Usage

```
get_dataset(dataset)
```

Arguments

dataset	Dataset URL or path.

Value

Object of type dataset_summary_response.

Examples

```
## Not run:
  dwapi::get_dataset(dataset = "user/dataset")

## End(Not run)
```

get_datasets_user_contributing *Search for datasets contributed-to by the currently authenticated user.*

Description

Search for datasets contributed-to by the currently authenticated user.

Usage

```
get_datasets_user_contributing(limit = NULL, next_page_token = NULL)
```

Arguments

limit	Maximum number of items to return
next_page_token	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named `records`, of objects of type `dataset_summary_response`. If the call to `get_datasets_user_contributing()` was made with a non-null `limit` parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named `next_page_token`, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  datasets_contributing <- dwapi::get_datasets_user_contributing()

## End(Not run)
```

---

get_datasets_user_liked

Search for datasets liked by the currently authenticated user.

Description

Search for datasets liked by the currently authenticated user.

Usage

```
get_datasets_user_liked(limit = NULL, next_page_token = NULL)
```

Arguments

`limit`	Maximum number of items to return
`next_page_token`	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named `records`, of objects of type `dataset_summary_response`. If the call to `get_datasets_user_liked()` was made with a non-null `limit` parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named `next_page_token`, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  datasets_liked <- dwapi::get_datasets_user_liked()

## End(Not run)
```

get_datasets_user_own *Search for datasets owned by the currently authenticated user.*

Description

Search for datasets owned by the currently authenticated user.

Usage

```
get_datasets_user_own(limit = NULL, next_page_token = NULL)
```

Arguments

limit	Maximum number of items to return
next_page_token	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named records, of objects of type dataset_summary_response. If the call to get_datasets_user_own() was made with a non-null limit parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named next_page_token, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  datasets_own <- dwapi::get_datasets_user_own()

## End(Not run)
```

get_insight *Retrieve an insight.*

Description

Retrieve an insight.

Usage

```
get_insight(project_owner, project_id, id)
```

Arguments

project_owner	username of the owner of the project.
project_id	identifier of the project.
id	identifier of the insight

Value

an object of type insight_summary_response.

Examples

```
## Not run:
  dwapi::get_insights(project_owner = "user", project_id = "project_id")

## End(Not run)
```

---

get_insights *Get insights for a project.*

Description

Get insights for a project.

Usage

```
get_insights(project_owner, project_id, limit = NULL,
  next_page_token = NULL)
```

Arguments

project_owner	username of the owner of the project.
project_id	identifier of the project.
limit	Maximum number of items to return
next_page_token	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named records, of objects of type insight_summary_response. If the call to get_insights() was made with a non-null limit parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named next_page_token, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  dwapi::get_insights(project_owner = "user", project_id = "project_id")

## End(Not run)
```

`get_project` *Retrieve a project.*

Description

Retrieve a project.

Usage

```
get_project(project_owner, project_id)
```

Arguments

`project_owner`	username of the owner of the project.
`project_id`	identifier of the project.

Value

an object of type `project_summary_response`.

Examples

```
## Not run:
  dwapi::get_project(project_owner = "user", project_id = "project_id")

## End(Not run)
```

`get_projects_user_contributing`
Search for projects contributed-to by the currently authenticated user.

Description

Search for projects contributed-to by the currently authenticated user.

Usage

```
get_projects_user_contributing(limit = NULL, next_page_token = NULL)
```

Arguments

limit	Maximum number of items to return
next_page_token	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named records, of objects of type project_summary_response. If the call to get_projects_user_contributing() was made with a non-null limit parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named next_page_token, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  projects_contributing <- dwapi::get_projects_user_contributing()

## End(Not run)
```

---

get_projects_user_liked

Search for projects liked by the currently authenticated user.

Description

Search for projects liked by the currently authenticated user.

Usage

```
get_projects_user_liked(limit = NULL, next_page_token = NULL)
```

Arguments

limit	Maximum number of items to return
next_page_token	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named records, of objects of type project_summary_response. If the call to get_projects_user_liked() was made with a non-null limit parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named next_page_token, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  projects_liked <- dwapi::get_projects_user_liked()

## End(Not run)
```

`get_projects_user_own` *Search for projects owned by the currently authenticated user.*

Description

Search for projects owned by the currently authenticated user.

Usage

```
get_projects_user_own(limit = NULL, next_page_token = NULL)
```

Arguments

`limit`	Maximum number of items to return
`next_page_token`	Unique token used to retrieve next page

Value

a named list with at most two elements. It will always contain a list, named `records`, of objects of type `project_summary_response`. If the call to `get_projects_user_own()` was made with a non-null `limit` parameter, and if further pages remain for retrieval, then the list will also contain a single-element character vector, named `next_page_token`, with the token to use in a subsequent call to get the next page.

Examples

```
## Not run:
  projects_own <- dwapi::get_projects_user_own()

## End(Not run)
```

get_table_schema *Retrieve schema information for a dataset table.*

Description

Retrieve schema information for a dataset table.

Usage

```
get_table_schema(dataset, table_name)
```

Arguments

dataset	Dataset URL or path.
table_name	Table name.

Value

Object of type table_schema_response.

See Also

list_tables

Examples

```
## Not run:
  table_schema <- dwapi::get_table_schema("user/dataset", "table")

## End(Not run)
```

get_user *Retrieve user information for the currently authenticated user.*

Description

Retrieve user information for the currently authenticated user.

Usage

```
get_user()
```

Value

Object of type user_info_response.

Examples

```
## Not run:
  my_dw_user <- dwapi::get_user()

## End(Not run)
```

`insight_create_request` *Create insight object for new insights*

Description

Create insight object for new insights

Usage

```
insight_create_request(title, description = NULL, image_url = NULL,
  embed_url = NULL, markdown_body = NULL, source_link = NULL,
  data_source_links = NULL)
```

Arguments

`title`	Insight title
`description`	Optional insight description
`image_url`	URL of image representing the insight
`embed_url`	URL of content to embed
`markdown_body`	Markdown text containing the insight
`source_link`	Permalink to source code or platform this insight was generated with
`data_source_links`	List containing one or more permalinks to the data sources used to generate this insight

Value

Request object of type `insight_create_request`.

See Also

`create_insight`

Examples

```
request <- dwapi::insight_create_request(title='A title',
  description = 'A description',
  image_url = 'https://site.org/image.png',
  source_link = 'https://site.org/data')
```

`insight_replace_request` *Create insight object for insights that replace existing insights*

Description

Create insight object for insights that replace existing insights

Usage

```
insight_replace_request(title, description = NULL, image_url = NULL,
  embed_url = NULL, markdown_body = NULL, source_link = NULL,
  data_source_links = NULL)
```

Arguments

`title`	Insight title
`description`	Optional insight description
`image_url`	URL of image representing the insight
`embed_url`	URL of content to embed
`markdown_body`	Markdown text containing the insight
`source_link`	Permalink to source code or platform this insight was generated with
`data_source_links`	List containing one or more permalinks to the data sources used to generate this insight

Value

Request object of type `insight_create_request`.

See Also

`create_insight`

Examples

```
request <- dwapi::insight_replace_request(title='A title',
  description = 'A description',
  image_url = 'https://site.org/image.png',
  source_link = 'https://site.org/data')
```

insight_summary_response

Deserialize `get_insight` *response object.*

Description

Deserialize `get_insight` response object.

Usage

```
insight_summary_response(structure)
```

Arguments

`structure`	httr response object.

Value

Object of type `insight_summary_response`.

See Also

`get_insights` `get_insight`

insight_update_request

Create insight object for updating existing insights

Description

Create insight object for updating existing insights

Usage

```
insight_update_request(title = NULL, description = NULL, image_url = NULL,
  embed_url = NULL, markdown_body = NULL, source_link = NULL,
  data_source_links = NULL)
```

Arguments

title	Insight title
description	Optional insight description
image_url	URL of image representing the insight
embed_url	URL of content to embed
markdown_body	Markdown text containing the insight
source_link	Permalink to source code or platform this insight was generated with
data_source_links	List containing one or more permalinks to the data sources used to generate this insight

Value

Request object of type `insight_replace_request`.

See Also

`create_insight`

Examples

```
request <- dwapi::insight_update_request(title='New title')
request <- dwapi::insight_update_request(title='New title',
    description = 'New description')
```

---

linked_dataset_create_or_update_request

Create request object for new linked datasets, or updates to existing datasets.

Description

Create request object for new linked datasets, or updates to existing datasets.

Usage

```
linked_dataset_create_or_update_request(owner, id)
```

Arguments

owner	User name and unique identifier of the creator of the dataset.
id	Unique identifier of dataset.

Value

Request object of type `linked_dataset_create_or_update_request`.

See Also

create_project

Examples

```
request <- dwapi::linked_dataset_create_or_update_request(
  owner = 'ownerId', id = 'datasetId')
```

`link_dataset` *Link a dataset to a project.*

Description

Link a dataset to a project.

Usage

```
link_dataset(project_owner, project_id, dataset_owner, dataset_id)
```

Arguments

`project_owner`	ID of project owner.
`project_id`	ID of project to which the dataset is to be linked.
`dataset_owner`	ID of dataset owner.
`dataset_id`	ID of dataset to be linked.

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::link_dataset(
    project_owner = 'user1',
    project_id = 'project',
    dataset_owner = 'user2',
    dataset_id = 'dataset')

## End(Not run)
```

list_tables *List tables in a dataset.*

Description

Tables are a logical concept representing normalized data extracted from tabular files uploaded to datasets on data.world.

Usage

```
list_tables(dataset)
```

Arguments

dataset	Dataset URL or path.

Details

EXPERIMENTAL: This is an experimental feature and backwards-compability is not guaranteed in future releases.

Value

list of table names.

Examples

```
## Not run:
  tables <- dwapi::list_tables("user/dataset")

## End(Not run)
```

parse_success_or_error *Parse simple responses (success or error).*

Description

Parse simple responses (success or error).

Usage

```
parse_success_or_error(response)
```

Arguments

response	httr response.

Value

Deserialized success or error.

`project_create_request` *Create request object for new projects.*

Description

Create request object for new projects.

Usage

```
project_create_request(title, visibility = c("PRIVATE", "OPEN"),
  objective = NULL, summary = NULL, tags = character(), license = NULL,
  files = list(), linked_datasets = list())
```

Arguments

`title`	Project title.
`visibility`	Project visibility, must be PRIVATE or OPEN.
`objective`	(optional) Short project objective.
`summary`	(optional) Long-form project summary (Markdown supported).
`tags`	Character vector of project tags (letters, numbers and spaces).
`license`	Dataset license ("Public Domain", "PDDL", "CC-0", "CC-BY", "ODC-BY", "CC-BY-SA", "ODC-ODbL", "CC BY-NC-SA" or Other).
`files`	(optional) List of `file_create_request` objects.
`linked_datasets`	List of `linked_dataset_create_or_update_request` objects.

Value

Request object of type `project_create_request`.

See Also

`create_project`

Examples

```
request <- dwapi::project_create_request(title='My Project', visibility = 'OPEN',
  objective = 'objective', tags = c('sdk') , license = 'Public Domain')
```

project_replace_request *Create request object for replacement of existing projects.*

Description

Create request object for replacement of existing projects.

Usage

```
project_replace_request(title, visibility = c("PRIVATE", "OPEN"),
  objective = NULL, summary = NULL, tags = character(), license = NULL,
  files = list(), linked_datasets = list())
```

Arguments

title	Project title.
visibility	Project visibility, must be PRIVATE or OPEN.
objective	(optional) Short project objective.
summary	(optional) Long-form project summary (Markdown supported).
tags	Character vector of project tags (letters, numbers and spaces).
license	Dataset license ("Public Domain", "PDDL", "CC-0", "CC-BY", "ODC-BY", "CC-BY-SA", "ODC-ODbL", "CC BY-NC-SA" or Other).
files	(optional) List of file_create_request objects.
linked_datasets	List of linked_dataset_create_or_update_request objects.

Value

Request object of type project_replace_request.

See Also

replace_project

Examples

```
request <- dwapi::project_replace_request(title='My Project', visibility = 'OPEN',
  objective = 'objective', tags = c('sdk') , license = 'Public Domain')
```

project_summary_response *Deserialize* get_project *response object.*

Description

Deserialize get_project response object.

Usage

```
project_summary_response(structure)
```

Arguments

structure	httr response object.

Value

Object of type project_summary_response.

See Also

get_projects_user_own

project_update_request *Create request object to update existing projects.*

Description

Create request object to update existing projects.

Usage

```
project_update_request(title = NULL, visibility = NULL, objective = NULL,
  summary = NULL, tags = NULL, license = NULL, files = NULL,
  linked_datasets = NULL)
```

Arguments

title	Project title.
visibility	Project visibility, must be PRIVATE or OPEN.
objective	(optional) Short project objective.
summary	(optional) Long-form project summary (Markdown supported).
tags	Character vector of project tags (letters, numbers and spaces).
license	Dataset license ("Public Domain", "PDDL", "CC-0", "CC-BY", "ODC-BY", "CC-BY-SA", "ODC-ODbL", "CC BY-NC-SA" or Other).
files	(optional) List of file_create_request objects.
linked_datasets	List of linked_dataset_create_or_update_request objects.

Value

Request object of type project_update_request.

See Also

update_project

Examples

```
request <- dwapi::project_update_request(title='My Project', visibility = 'OPEN',
  objective = 'objective', tags = c('sdk') , license = 'Public Domain')
```

---

replace_dataset *Replace an existing dataset.*

Description

Replace an existing dataset.

Usage

```
replace_dataset(dataset, dataset_replace_req)
```

Arguments

dataset	Dataset URL or path.
dataset_replace_req	Request object of type dataset_replace_request.

Value

Object of type success_message.

Examples

```
dataset_replace_req <- dwapi::dataset_replace_request(visibility = 'OPEN',
 description = 'UPDATED DESCRIPTION !', title = 'Updated Title')
## Not run:
  dwapi::replace_dataset('user/dataset', dataset_replace_req)

## End(Not run)
```

`replace_insight` *Replace an insight.*

Description

Replace an insight.

Usage

```
replace_insight(project_owner, project_id, id, replace_insight_req)
```

Arguments

`project_owner`	ID of project owner
`project_id`	ID of project to which insight is to be added
`id`	ID of insight to be replaced
`replace_insight_req`	Request object of type `insight_replace_request`.

Value

Object of type `success_message`.

Examples

```
## Not run:
  dwapi::replace_insight(
    project_owner = 'user',
    project_id = 'project',
    id = 'insight',
    insight_replace_request(title='My Insight', image_url='https://...'))

## End(Not run)
```

`replace_project` *Create a new project with a specific ID, replacing a project with that ID if it exists.*

Description

Create a new project with a specific ID, replacing a project with that ID if it exists.

Usage

```
replace_project(owner_id, project_id, replace_project_req)
```

Arguments

`owner_id`	data.world user name of the project owner.
`project_id`	identifier to use for the new project, or identifier of existing project to replace with the newly created project.
`replace_project_req`	Request object of type `project_replace_request`.

Value

Object of type `success_message`.

Examples

```
request <- dwapi::project_replace_request(
  title='testproject', visibility = 'OPEN',
  objective = 'Test project by R-SDK', tags = c('rsdk', 'sdk', 'arr'),
  license = 'Public Domain')

request <- dwapi::add_file(request = request, name = 'file4.csv',
  url = 'https://data.world/file4.csv')

## Not run:
dwapi::replace_project(create_project_req = request,
  owner_id = 'user', project_id = 'projectid')

## End(Not run)
```

sdk_version *Return the current dwapi version*

Description

Return the current dwapi version

Usage

```
sdk_version()
```

Value

Current package version

sparql *Execute SPARQL query against a dataset.*

Description

#' **EXPERIMENTAL**: This is an experimental feature and backwards-compability is not guaranteed in future releases.

Usage

```
sparql(dataset, query, query_params = list())
```

Arguments

dataset	Dataset URL or path.
query	SPARQL query.
query_params	List of named query parameters.

Value

Data frame with data from query results.

Examples

```
## Not run:
 dwapi::sparql(dataset="user/dataset",
   query="SELECT *
          WHERE {
            ?s ?p ?o .
          } LIMIT 10")

dwapi::sparql(dataset="user/dataset",
  query="SELECT *
         WHERE {
        [ :Year ?year ; :Region ?region ; :Indicator_Coverage_and_Disaggregation ?score ]
         FILTER(?score > $v1)
         } LIMIT 10",
  queryParameters = list("$v1"=5.5))

## End(Not run)
```

`sql` *Execute SQL query against a dataset.*

Description

Execute SQL query against a dataset.

Usage

```
sql(dataset, query, query_params = list())
```

Arguments

`dataset`	Dataset URL or path.
`query`	SQL query.
`query_params`	List of positional query parameters.

Value

Data frame with data from query results.

Examples

```
## Not run:
  dwapi::sql(dataset="user/dataset",
    query="SELECT *
           FROM TableName
           LIMIT 10")

  dwapi::sql(dataset="user/dataset",
```

```
    query="SELECT *
           FROM TableName where `field1` = ? AND `field2` > ?
           LIMIT 10",
    queryParameters = list("value", 5.0))

## End(Not run)
```

`success_message` *Deserialize success response object.*

Description

Deserialize success response object.

Usage

```
success_message(structure)
```

Arguments

`structure`	httr response object

Value

Object of type `success_message`

`sync` *Fetch latest files from source and update dataset.*

Description

Fetch latest files from source and update dataset.

Usage

```
sync(dataset)
```

Arguments

`dataset`	Dataset URL or path.

Value

Object of type `success_message`.

Examples

```
## Not run:
  dwapi::sync(dataset='user/dataset')

## End(Not run)
```

`table_schema_field_response` *Deserialize* `get_table_schema` *fields.*

Description

Deserialize `get_table_schema` fields.

Usage

```
table_schema_field_response(structure)
```

Arguments

`structure`	httr response object

Value

Object of type `table_schema_field_response`

`table_schema_field_update_request` *Create request object for updating table schema fields.*

Description

Create request object for updating table schema fields.

Usage

```
table_schema_field_update_request(name, description)
```

Arguments

`name`	Table field name.
`description`	Table field description.

Value

Object of type `table_schema_field_update_request`

Examples

```
field_update_req <- dwapi::table_schema_field_update_request("field", "new desc")
```

---

table_schema_response *Deserialize* get_table_schema *response object.*

Description

Deserialize get_table_schema response object.

Usage

```
table_schema_response(structure)
```

Arguments

structure	httr response object.

Value

Object of type table_schema_response

See Also

get_table_schema

---

table_schema_update_request *Create request object for updating table schema.*

Description

Create request object for updating table schema.

Usage

```
table_schema_update_request(fields)
```

Arguments

fields	List of table_schema_field_update_request objects.

Value

Object of type table_schema_update_request.

See Also

update_table_schema

Examples

```
field_update_req <- dwapi::table_schema_field_update_request("field", "new desc")
schema_update_req <- dwapi::table_schema_update_request(c(field_update_req))
```

---

unlink_dataset *Unlink a dataset from a project.*

Description

Unlink a dataset from a project.

Usage

```
unlink_dataset(project_owner, project_id, dataset_owner, dataset_id)
```

Arguments

project_owner	ID of project owner.
project_id	ID of project from which the dataset is to be unlinked.
dataset_owner	ID of dataset owner.
dataset_id	ID of dataset to be unlinked.

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::unlink_dataset(
    project_owner = 'user1',
    project_id = 'project',
    dataset_owner = 'user2',
    dataset_id = 'dataset')

## End(Not run)
```

update_dataset *Update an existing dataset.*

Description

Update an existing dataset.

Usage

```
update_dataset(dataset, dataset_update_req)
```

Arguments

dataset	Dataset URL or path.
dataset_update_req	Request object of type dataset_update_request.

Value

Object of type success_message.

Examples

```
request <- dwapi::dataset_update_request(visibility = 'OPEN',
  description = 'UPDATED DESCRIPTION !')

## Not run:
  dwapi::update_dataset(dataset_update_req = request,
    dataset = 'user/dataset')

## End(Not run)
```

update_insight *Update an insight.*

Description

Update an insight.

Usage

```
update_insight(project_owner, project_id, id, update_insight_req)
```

Arguments

project_owner	ID of project owner
project_id	ID of project of which insight is a component
id	ID of insight to be replaced
update_insight_req	Request object of type insight_update_request.

Value

Object of type success_message.

Examples

```
## Not run:
  dwapi::update_insight(
    project_owner = 'user',
    project_id = 'project',
    id = 'insight',
    insight_update_request(title='My Insight', image_url='https://...'))

## End(Not run)
```

---

update_project *Update an existing project.*

Description

Update an existing project.

Usage

```
update_project(project, project_update_req)
```

Arguments

project	Project URL or path.
project_update_req	Request object of type project_update_request.

Value

Object of type success_message.

Examples

```
request <- dwapi::project_update_request(
  objective = 'UPDATED OBJECTIVE !')

## Not run:
  dwapi::update_project(project_update_req = request,
    dataset = 'user/project')

## End(Not run)
```

`update_table_schema` *Update table schema.*

Description

Update table schema.

Usage

```
update_table_schema(dataset, table_name, table_schema_update_req)
```

Arguments

`dataset`	Dataset URL or path.
`table_name`	Table name.
`table_schema_update_req`	Request object of type `table_schema_update_request`

Value

Object of type `success_message`

Examples

```
field_update_req <- dwapi::table_schema_field_update_request("field", "new desc")
schema_update_req <- dwapi::table_schema_update_request(c(field_update_req))
## Not run:
  dwapi::update_table_schema("user/dataset", "table", schema_update_req)

## End(Not run)
```

upload_data_frame *Upload a data frame as a file to a dataset.*

Description

Upload a data frame as a file to a dataset.

Usage

```
upload_data_frame(dataset, data_frame, file_name)
```

Arguments

dataset	Dataset URL or path.
data_frame	Data frame object.
file_name	File name, including file extension.

Value

Server response message.

Examples

```
df = data.frame(a = c(1,2,3),b = c(4,5,6))
## Not run:
  dwapi::upload_data_frame(file_name = 'sample.csv',
    data_frame = df, dataset = 'user/dataset')

## End(Not run)
```

upload_file *Upload a single file to a dataset.*

Description

Upload a single file to a dataset.

Usage

```
upload_file(dataset, path, file_name)
```

Arguments

dataset	Dataset URL or path.
path	File path on local file system.
file_name	File name, including file extension.

Value

Server response message.

Examples

```
## Not run:
  dwapi::upload_file(file_name = 'file.csv',
    path = 'file.csv', dataset = 'user/dataset')

## End(Not run)
```

---

upload_files *Upload one or more files to a dataset.*

Description

Upload one or more files to a dataset.

Usage

```
upload_files(dataset, paths)
```

Arguments

dataset	Dataset URL or path.
paths	List of file paths on local file system.

Value

Server response message.

Examples

```
## Not run:
  dwapi::upload_files(dataset = 'user/dataset',
    paths = c('file1.csv', 'file2.csv'))

## End(Not run)
```

user_agent *Return the dwapi user-agent*

Description

Return the dwapi user-agent

Usage

```
user_agent()
```

Value

User-agent string

user_info_response *Deserialize* get_user *response object.*

Description

Deserialize get_user response object.

Usage

```
user_info_response(structure)
```

Arguments

structure	httr response object.

Value

Object of type user_info_response.

See Also

get_dataset

Index

add_file, 4, *16, 17*, *19*, *29*
add_file_by_source, 6
add_files_by_source, 5, *26*, *29*
append_data_frame_to_stream, 7, *26*
append_record_to_stream, 8, *26*
append_values_to_stream, 9, *26*
auth_token, 10

check_file_summary_response, 10
check_project_summary_response, 11
check_user_info_response, 11
configure, 12, *25*
create_dataset, 12, *16*, *25*, *31*
create_dataset_response, *13*, 13
create_insight, 14, *26*, *41, 42*, *44*
create_insight_response, *14*
create_project, 14, *26*, *45*, *47*
create_project_response, *15*, 15

dataset_create_request, *12*, 16, *27*
dataset_replace_request, 17, *27*, *50*
dataset_summary_response, *18*, 18, *33–35*
dataset_update_request, 18, *27*, *59*
delete_dataset, 19, *25*
delete_file, 20, *26*
delete_files, 20
delete_insight, 21, *26*
delete_project, 22, *26*
download_datapackage, 22, *25*
download_file, 23, *25*
download_file_as_data_frame, 24, *25*
download_table_as_data_frame, 24, *26*
dwapi, 25
dwapi-package *(*dwapi*)*, 25

error_message, *27*, 27
extract_dataset_key, 28
extract_project_key, 28

file_batch_update_request, *5*, *27*, 29, *30*
file_create_or_update_request, *18*, *27*, *29*, 29
file_create_request, *16, 17*, *27*, 30, *47, 48*, *50*
file_source_create_or_update_request, *27*, 31
file_source_create_request, *27*, 31
file_source_summary_response, 32
file_summary_response, *10*, 32

get_dataset, *18*, *25*, 33, *64*
get_datasets_user_contributing, 33
get_datasets_user_liked, 34
get_datasets_user_own, 35
get_insight, *26*, 35, *43*
get_insights, *26*, 36, *43*
get_project, *26*, 37
get_projects_user_contributing, *26*, 37
get_projects_user_liked, *26*, 38
get_projects_user_own, *26*, 39, *49*
get_table_schema, *26*, 40, *57*
get_user, 40

insight_create_request, *14*, *27*, 41
insight_replace_request, *27*, 42, *51*
insight_summary_response, *43*, 43
insight_update_request, *27*, 43, *60*

link_dataset, 45
linked_dataset_create_or_update_request, *27*, 44, *47, 48*, *50*
list_tables, *25, 26*, *40*, 46

parse_success_or_error, 46
project_create_request, *15*, *27*, 47
project_replace_request, *27*, 48, *52*
project_summary_response, *11*, *38, 39*, *49*, 49
project_update_request, *27*, 49, *60*

replace_dataset, *17*, *25*, *31*, 50

replace_insight, *26*, 51
replace_project, *26*, *48*, 52
RETRY, *7–9*

sdk_version, 53
sparql, *26*, 53
sql, *26*, 54
success_message, *5, 6*, *19–22*, *45*, *50–52*, *55*, 55, *58–61*
sync, *25*, 55

table_schema_field_response, 56
table_schema_field_update_request, 56, *57*
table_schema_response, *40*, 57
table_schema_update_request, *27*, 57, *61*

unlink_dataset, 58
update_dataset, *19*, *25*, *30*, 59
update_insight, *26*, 59
update_project, *26*, *50*, 60
update_table_schema, *26*, *58*, 61
upload_data_frame, *25*, 62
upload_file, *25*, 62
upload_files, 63
user_agent, 64
user_info_response, *11*, *40*, *64*, 64