Package 'dwapi'

January 11, 2018

Title A Client for 'data.world' REST API

Version 0.1.2

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

Suggests knitr, rmarkdown, testthat

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]

Maintainer Rafael Pereira <rafael.pereira@data.world>

Repository CRAN

Date/Publication 2018-01-10 23:37:15 UTC

R topics documented:

`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', 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)
```

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_dataset_summary_response *Validate* get_dataset *response object.*

Description

Validate get_dataset response object.

Usage

```
check_dataset_summary_response(object)
```

Arguments

object	Object of type dataset_summary_response.

Value

TRUE/FALSE based on validity of object.

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)
```

`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(description = NULL, summary = NULL, tags = NULL,
  license_string = NULL, visibility, 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	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')
```

`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_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)
```

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. sync
6. 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 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. table_schema_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

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_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')
```

`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_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

---

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 !')
## Not run:
  dwapi::replace_dataset('user/dataset', dataset_replace_req)

## 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))
```

---

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_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