

Package ‘viafoundry’

July 22, 2025

Type Package

Title R Client for 'Via Foundry' API

Version 1.0.0

Language en-US

Description 'Via Foundry' API provides streamlined tools for interacting with and extracting data from structured responses, particularly for use cases involving hierarchical data from Foundry's API. It includes functions to fetch and parse process-level and file-level metadata, allowing users to efficiently query and manipulate nested data structures. Key features include the ability to list all unique process names, retrieve file metadata for specific or all processes, and dynamically load or download files based on their type. With built-in support for handling various file formats (e.g., tabular and non-tabular files) and seamless integration with API through authentication, this package is designed to enhance workflows involving large-scale data management and analysis. Robust error handling and flexible configuration ensure reliable performance across diverse data environments. Please consult the documentation for the API endpoint for your installation.

License Apache License 2.0

Encoding UTF-8

Imports httr, jsonlite, dplyr, purrr, askpass

RoxygenNote 7.3.2

URL <https://github.com/ViaScientific/viafoundry-R-SDK>

NeedsCompilation no

Author Alper Kucukural [aut, cre],
Via Scientific [aut, cph]

Maintainer Alper Kucukural <alper@viascientific.com>

Repository CRAN

Date/Publication 2025-01-13 18:00:02 UTC

Contents

authenticate	2
calculate_expiration_date	3

call_endpoint	3
discover	4
fetchReportData	4
getAllFileNames	5
getFileNames	5
getProcessNames	6
get_api_status	6
get_bearer_token	7
get_headers	7
loadFile	8
load_config	8
login	9

Index	10
--------------	-----------

authenticate	<i>Authenticate with the ‘Via Foundry’ API</i>
--------------	--

Description

Authenticates the user with the ‘Via Foundry’ API using their username and password. Retrieves a bearer token and saves it along with the hostname to a configuration file.

Usage

```
authenticate(  
    hostname,  
    username = NULL,  
    password = NULL,  
    identity_type = 1,  
    redirect_uri = "http://localhost",  
    config_path = DEFAULT_CONFIG_PATH,  
    overwrite = FALSE  
)
```

Arguments

hostname	The API ‘hostname’ (e.g., ‘https://your_viafoundry’).
username	The login username.
password	The login password (optional; will prompt if not provided).
identity_type	The identity type (default is 1).
redirect_uri	The redirect ‘URI’ (default is ‘http://your_viafoundry/redirect’).
config_path	Path to save the configuration file (default is ‘~/.viaenv’).
overwrite	Logical flag to overwrite the existing configuration file (default is FALSE).

Value

None. Saves the bearer token to the configuration file and sets the global config path.

calculate_expiration_date	<i>Calculate expiration date (30 days from now)</i>
---------------------------	---

Description

Calculate expiration date (30 days from now)

Usage

```
calculate_expiration_date()
```

Value

The expiration date as a string.

call_endpoint	<i>Call an API Endpoint</i>
---------------	-----------------------------

Description

Sends an HTTP request to a specified API endpoint using the stored bearer token for authentication.

Usage

```
call_endpoint(method, endpoint, params = list(), data = NULL)
```

Arguments

method	The HTTP method (e.g., "GET", "POST", "PUT", "DELETE").
endpoint	The API endpoint (e.g., '/api/projects').
params	A named list of query parameters (optional).
data	A named list or 'JSON' object to include in the request body (optional).

Value

A list containing the API response.

Examples

```
## Not run:  
response <- call_endpoint("GET", "/api/projects")  
print(response)  
  
## End(Not run)
```

discover*Discover Available Endpoints*

Description

Fetches and lists all available API endpoints from the Swagger documentation.

Usage

```
discover()
```

Value

A character vector of available endpoints.

Examples

```
## Not run:  
discover()  
  
## End(Not run)
```

fetchReportData*Fetch the 'JSON' data for a report*

Description

Fetch the 'JSON' data for a report

Usage

```
fetchReportData(reportID)
```

Arguments

reportID The ID of the report to fetch data for.

Value

The 'JSON' object containing the report data.

getAllFileNames	<i>Extract children names across all processes</i>
-----------------	--

Description

Extract children names across all processes

Usage

```
getAllFileNames(json_data)
```

Arguments

json_data	The 'JSON' object containing the report data.
-----------	---

Value

A data frame containing 'id', 'processName', 'name', 'extension', 'fileSize', and 'routePath'.

getFileNames	<i>Get file names for a specific process</i>
--------------	--

Description

Get file names for a specific process

Usage

```
getFileNames(json_data, processName)
```

Arguments

json_data	The 'JSON' object containing the report data.
processName	The name of the process to filter by.

Value

A data frame containing 'id', 'name', 'extension', 'fileSize', and 'routePath'.

getProcessNames	<i>Get unique process names</i>
-----------------	---------------------------------

Description

Get unique process names

Usage

```
getProcessNames(json_data)
```

Arguments

json_data The 'JSON' object containing the report data.

Value

A character vector of unique process names.

get_api_status	<i>Get API Status</i>
----------------	-----------------------

Description

Sends a simple GET request to check the status of the API.

Usage

```
get_api_status()
```

Value

A character vector with the API status.

Examples

```
## Not run:  
status <- get_api_status()  
print(status)  
  
## End(Not run)
```

get_bearer_token	<i>Get bearer token using the cookie token</i>
------------------	--

Description

Get bearer token using the cookie token

Usage

```
get_bearer_token(hostname, cookie_token, name = "token")
```

Arguments

hostname	The API hostname.
cookie_token	The cookie token.
name	The name of the token (default is "token").

Value

The bearer token.

get_headers	<i>Get headers for API requests</i>
-------------	-------------------------------------

Description

Get headers for API requests

Usage

```
get_headers()
```

Value

A list of headers with the bearer token.

loadFile	<i>Load or download a file from a process and file name</i>
----------	---

Description

Load or download a file from a process and file name

Usage

```
loadFile(json_data, processName, fileName, sep = "\t", download_dir = getwd())
```

Arguments

json_data	The 'JSON' object containing the report data.
processName	The name of the process.
fileName	The name of the file to load or download.
sep	The separator for tabular files. Default is tab-separated.
download_dir	The directory where non-tabular files will be downloaded.

Value

A data frame with the file contents if the file is tabular; otherwise, NULL after downloading the file.

load_config	<i>Load Configuration</i>
-------------	---------------------------

Description

Loads the configuration file or triggers authentication if the file is missing.

Usage

```
load_config()
```

Value

A list containing the hostname and bearer token.

`login`

Login and retrieve the cookie token

Description

Login and retrieve the cookie token

Usage

```
login(  
    hostname,  
    username,  
    password,  
    identity_type = 1,  
    redirect_uri = "http://localhost"  
)
```

Arguments

<code>hostname</code>	The API hostname.
<code>username</code>	The login username.
<code>password</code>	The login password.
<code>identity_type</code>	The identity type.
<code>redirect_uri</code>	The redirect URI.

Value

The cookie token.

Index

authenticate, [2](#)

calculate_expiration_date, [3](#)

call_endpoint, [3](#)

discover, [4](#)

fetchReportData, [4](#)

get_api_status, [6](#)

get_bearer_token, [7](#)

get_headers, [7](#)

getAllFileNames, [5](#)

getFileNames, [5](#)

getProcessNames, [6](#)

load_config, [8](#)

loadFile, [8](#)

login, [9](#)