Package ‘metricminer’

February 2, 2024

Type    Package
Title   Mine Metrics from Common Places on the Web
Version 0.5.1
Description Mine metrics on common places on the web through the power of their APIs (application programming interfaces).
It also helps make the data in a format that is easily used for a dashboard or other purposes.
There is an associated dashboard template and tutorials that are underdevelopment that help you fully utilize ‘metricminer’.
License GPL-3
URL https://github.com/fhdsl/metricminer
BugReports https://github.com/fhdsl/metricminer/issues
Imports httr, jsonlite, assertthat, openssl, gh, getPass, dplyr,
lubridate, purrr, tidyr, googledrive, googlesheets4, janitor,
stringr, methods,
Suggests knitr, rmarkdown, testthat (>= 3.0.0), withr
Config/testthat/edition 3
Encoding UTF-8
RoxygenNote 7.2.3
VignetteBuilder knitr
NeedsCompilation no
Author Candace Savonen [aut, cre],
       Howard Baek [aut]
Maintainer Candace Savonen <cansav09@gmail.com>
Depends R (>= 3.5.0)
Repository CRAN
Date/Publication 2024-02-02 19:20:02 UTC
R topics documented:

app_set_up ................................................................. 3
authorize ................................................................. 3
auth_from_secret ......................................................... 4
calendly_get ............................................................. 5
check_check ............................................................... 6
clean_ga_metrics ......................................................... 7
clean_repo_metrics ....................................................... 7
default_creds_path ...................................................... 8
delete_creds ............................................................... 8
encrypt_creds_path ....................................................... 9
example_data_folder .................................................... 9
extract_answers ........................................................... 9
get_all_ga_metrics ....................................................... 10
get_calendly_user ....................................................... 11
get_example_data ........................................................ 11
get_ga_metadata ........................................................ 12
get_ga_properties ......................................................... 13
get_ga_stats ............................................................... 13
get_ga_user ............................................................... 15
github ...................................................................... 15
get_github ............................................................... 15
get_github_metrics ....................................................... 16
get_github_repo_summary ............................................... 17
get_github_repo_timecourse .......................................... 18
get_github_user ........................................................ 19
get_google_form ........................................................ 19
get_multiple_forms ...................................................... 20
get_multiple_repos_metrics ........................................... 21
get_org_repo_list ........................................................ 22
get_question_metadata ................................................ 23
get_slido_files .......................................................... 23
get_timestamp_repo_metrics ......................................... 24
get_user_repo_list ....................................................... 24
get_youtube_channel_stats .......................................... 25
google_form .............................................................. 25
get_youtube_video_stats .............................................. 26
gh_repo_wrapper ........................................................ 27
key_encrypt_creds_path ............................................... 27
list_calendly_events .................................................... 28
list_example_data ........................................................ 28
request_ga .............................................................. 29
request_google_forms ................................................... 30
supported_endpoints .................................................... 30
write_playlist_details ................................................ 31
write_to_gsheet ........................................................ 31

Index 34
**app_set_up**

**App Set Up**

**Description**

This is a function that sets up the app. It’s generally called by another function.

**Usage**

```r
app_set_up(app_name = "google")
```

**Arguments**

- `app_name`: app would you like to authorize? Supported apps are ‘google’ ‘calendly’ and ‘github’

**authorize**

**Authorize R package to access endpoints**

**Description**

This is a function to authorize the R package to access APIs interactively.

**Usage**

```r
authorize(app_name = NULL, cache = FALSE, ...)
```

**Arguments**

- `app_name`: app would you like to authorize? Supported apps are ‘google’ ‘calendly’ and ‘github’
- `cache`: Should the token be cached as an .httr-oauth file or API keys stored as global options?
- `...`: additional arguments to send to `oauth2_0_token`

**Value**

API token saved to the environment or the cache so it can be grabbed by functions.
auth_from_secret

**Examples**

```r
## Not run:
authorize()
authorize("github")
authorize("google")
authorize("calendly")
## End(Not run)
```

**Description**

This is a function to authorize metricminer to access calendly, github or google noninteractively from passing in a keys or tokens.

**Usage**

```r
auth_from_secret(
  app_name,
  token,
  access_token,
  refresh_token,
  cache = FALSE,
  in_test = TRUE
)
```

**Arguments**

- **app_name**: Which app are you trying to authorize? 'google', 'calendly' or 'github'?  
- **token**: For calendly or github, pass in the API key or Personal Access Token that you have set up from going to https://github.com/settings/tokens/new or https://calendly.com/integrations/api_webhooks respectively. 
- **access_token**: For Google, access token can be obtained from running authorize interactively: token <-authorize(); token$credentials$access_token 
- **refresh_token**: For Google, refresh token can be obtained from running authorize interactively: token <-authorize(); token$credentials$refresh_token 
- **cache**: Should the credentials be cached? TRUE or FALSE? 
- **in_test**: If setting up auth in a test, set to TRUE so that way the authorization doesn’t stick
Value

OAuth token saved to the environment so the package access the API data

Examples

```r
## Not run:

# Example for authorizing Calendly
# You go to https://calendly.com/integrations/api_webhooks to get an api key
auth_from_secret("calendly", token = "A_calendly_token_here")

# Example for GitHub
# You go to https://github.com/settings/tokens/new to get a Personal Access Token
auth_from_secret("github", token = "ghp_a_github_pat_here")

# Example for authorizing for Google
token <- authorize("google")
auth_from_secret(
  app_name = "google",
  access_token = token$credentials$access_token,
  refresh_token = token$credentials$refresh_token
)

## End(Not run)
```

---

calendly_get  

*Handle Calendly GET requests*

Description

This is a function that handles Calendly GET requests

Usage

```r
calendly_get(url, token = NULL, user = NULL, count = NULL, page_token = NULL)
```

Arguments

- **url**: The endpoint URL for this API request
- **token**: You can provide the API key directly using this argument or this function will attempt to grab an API key that was stored using the `authorize("calendly")` function
- **user**: The user param for Calendly. Usually looks like "https://api.calendly.com/users/c208a750-9214-4c62-9ee6-a1a9507c7b43"
- **count**: For paginated GETs, you can specify how many things you’d like returned
- **page_token**: For a paginated GET, what page are we on?
check_check

Value

Calendly REST API response as a list

Examples

```r
## Not run:

authorize("calendly")
token <- get_token(app_name = "calendly")

result_list <- calendly_get(
  url = "https://api.calendly.com/users/me",
  token = token
)

## End(Not run)
```

---

**check_check**

*Check the testthat check log file and print out how many errors*

Description

if testthat’s tests have been run, this will look for the check to see if anything truly broke It will return a TRUE/FALSE for whether or not there were errors based on the check/testthat.Rout file produced.

Usage

```r
check_check(report_warning = TRUE)
```

Arguments

- `report_warning` Should the number include warnings in addition errors? Default is both will be reported but if you’d like to ignore warnings set this to FALSE.

Value

- a how many errors/warnings were found
clean_ga_metrics  Handle Google Analytics Lists

Description
These functions are to clean metric and dimension data from Google Analytics ‘get ga stats()’ function

Usage
```r
clean_ga_metrics(metrics = NULL)
```

Arguments
- **metrics** a metrics object from ‘get ga stats()’ function

Value
a data frame of cleaned metrics from Google Analytics

clean_repo_metrics  Summarizing metrics from GitHub

Description
This is a function to get metrics for all the repositories underneath an organization

Usage
```r
clean_repo_metrics(repo_name, repo_metric_list)
```

Arguments
- **repo_name** The repository name. So for ‘https://github.com/fhdsl/metricminer’, it would be ‘metricminer’
- **repo_metric_list** a list containing the metrics

Value
Metrics for a repository on GitHub
**default_creds_path**  
*Default Credentials path*

**Description**
Default Credentials path

**Usage**
```
default_creds_path(app_name)
```

**Arguments**
- `app_name`: What app setup are you looking for? Supported apps are ‘google’ ‘calendly’ and ‘github’ Get file path to an default credentials RDS

**delete_creds**  
*Delete cached metricminer credentials*

**Description**
This is a function to delete cached creds and creds in the current environment that were set by metricminer

**Usage**
```
delete_creds(app_name = "all")
```

**Arguments**
- `app_name`: which app would you like to delete the creds for? Default is to delete the creds for all.

**Value**
Cached credentials are deleted and report is given back

**Examples**
```
## Not run:
delete_creds("google")
## End(Not run)
```
encrypt_creds_path  Default creds path

Description
Default creds path

Usage
encrypt_creds_path(app_name)

Arguments
app_name  What app set up are you looking for? Supported apps are 'google' 'calendly' and 'github'

example_data_folder  Default Credentials path Get file path to an default credentials RDS

Description
Default Credentials path Get file path to an default credentials RDS

Usage
example_data_folder()

Value
Returns the file path to folder where the example data is stored

extract_answers  Google Form handling functions – extracting answers

Description
This is a function to get extract answers from a Google Form. It is used by the 'get_google_form()' function if dataformat = "dataframe"

Usage
extract_answers(form_info)
get_all_ga_metrics

Arguments

form_info The return form_info list that is extracted in `get_google_form()`

Value

This returns answers from a google form

get_all_ga_metrics Get all metrics for all properties associated with an account

Description

This is a function to get metrics and dimensions for all properties associated with an account

Usage

get_all_ga_metrics(account_id = NULL, token = NULL, dataformat = "dataframe")

Arguments

account_id the account id that you’d like to retrieve stats for all properties associated with it.
token credentials for access to Google using OAuth. `authorize("google")`
dataformat How would you like the data returned to you? Default is a "dataframe" but if you’d like to see the original API list result, put "raw".

Value

Either a list of dataframes where ‘metrics’, ‘dimensions’ and ‘link clicks’ are reported. But if ‘format’ is set to “raw” then the original raw API results will be returned

A list of metrics, dimensions, and link clicks for a for all properties underneath a Google Analytics account. It can be returned as a curated data.frame or the raw version which is the API response as a list

Examples

```r
## Not run:
authorize("google")
accounts <- get_ga_user()
some_stats_list <- get_all_ga_metrics(property_ids = property_ids)
## End(Not run)
```
get_calendly_user  Get Calendly API user

Description

This is a function to get the Calendly API user info.

Usage

get_calendly_user(token = NULL)

Arguments

- token: You can provide the API key directly using this argument or this function will attempt to grab an API key that was stored using the `authorize("calendly")` function.

Value

Calendly API user info as a list

Examples

```r
## Not run:
authorize("calendly")
get_calendly_user()

## End(Not run)
```

get_example_data  Get retrieve an example dataset

Description

This is a function to retrieve a list of the example datasets included with metricminer.

Usage

get_example_data(dataset_name, envir = 1)

Arguments

- dataset_name: the name of the example dataset to be retrieved from the metricminer package.
- envir: By default the example data is saved in the global environment but this parameter allows you to change that if desired.
get_ga_metadata

Value

an object in the environment of the same example dataset name that was requested.

Examples

## Not run:

# You can see the list of example datasets by running:
list_example_data()

# Then use the datasetes of your interest by calling it with this function
get_example_data("gform_info")

# Then if you check your global environment you will see "gform_info" included
ls()

## End(Not run)

get_ga_metadata

Get metadata associated Google Analytics property

Description

This is a function to get the Google Analytics accounts that this user has access to

Usage

get_ga_metadata(property_id, token = NULL)

Arguments

property_id a GA property. Looks like '123456789' Can be obtained from running 'get_ga_properties()' 
token credentials for access to Google using OAuth. 'authorize("google")'

Value

A list showing the metadata types available for the Google Analytics property. This can be used to craft an API request.

Examples

## Not run:

authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[1])

property_id <- gsub("properties/", "", properties_list$name[1])
get_ga_properties

property_metadata <- get_ga_metadata(property_id = property_id)

## End(Not run)

get_ga_properties

Get all property ids for all Google Analytics associated with an account id

Description

This is a function to get the Google Analytics accounts that this user has access to

Usage

get_ga_properties(account_id, token = NULL)

Arguments

account_id the account id of the properties you are trying to retrieve
token credentials for access to Google using OAuth. ‘authorize("google")’

Value

All the property ids and information about them for a Google Analytics account.

Examples

## Not run:
authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[1])

## End(Not run)

get_ga_stats

Get stats for an associated Google Analytics property

Description

This is a function to get the Google Analytics accounts that this user has access to
Usage

get_ga_stats(
  property_id,
  start_date = "2015-08-14",
  token = NULL,
  body_params = NULL,
  end_date = NULL,
  stats_type = "metrics",
  dataformat = "dataframe"
)

Arguments

property_id: a GA property. Looks like '123456789' Can be obtained from running `get_ga_properties()`

start_date: YYYY-MM-DD format of what metric you’d like to collect metrics from to start. Default is the earliest date Google Analytics were collected.

token: credentials for access to Google using OAuth. `authorize("google")`

body_params: The body parameters for the request

derDate: YYYY-MM-DD format of what metric you’d like to collect metrics from to end. Default is today.

stats_type: Do you want to retrieve metrics or dimensions?

dataformat: How would you like the data returned to you? Default is a "dataframe" but if you’d like to see the original API list result, put "raw".

Value

Metrics dimensions for a GA returned from the Google Analytics API. It can be returned as a curated data.frame or the raw version which is the API response as a list

Examples

```r
## Not run:
authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[1])

property_id <- gsub("properties/", "", properties_list$name[1])
metrics <- get_ga_stats(property_id, stats_type = "metrics")
dimensions <- get_ga_stats(property_id, stats_type = "dimensions")

## End(Not run)
```
get_\_ga\_user

Get Google Analytics Accounts

Description
This is a function to get the Google Analytics accounts that this user has access to

Usage
get_\_ga\_user(token = NULL, request\_type = "GET")

Arguments
- **token**: credentials for access to Google using OAuth. ‘authorize("google")’
- **request\_type**: Is this a GET or a POST?

Value
Information about what accounts Google Analytics credentials has access to

Examples
```
## Not run:
authorize("google")
get_\_ga\_user()
## End(Not run)
```

get_\_github

Handler function for GET requests from GitHub

Description
This is a function to get the GitHub user’s info

Usage
get_\_github(token = NULL, url)

Arguments
- **token**: You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")’ function
- **url**: What is the URL endpoint we are attempting to grab here?
get_github_metrics

Value

Information regarding a Github account

get_github_metrics  Get the repository summary or time course metrics

Description

This is a function to get the information about a repository

Usage

get_github_metrics(
  repo,
  token = NULL,
  count = "all",
  data_format = "dataframe",
  time_course = FALSE
)

Arguments

repo  The repository name. So for 'https://github.com/fhdsl/metricminer', it would be 'fhdsl/metricminer'
token  You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function
count  How many items would you like to receive? Put "all" to retrieve all records.
data_format  Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".
time_course  Should the time course data be collected or only the summary metrics?

Value

Repository summary or time course metrics for a particular GitHub repository as a dataframe

Examples

## Not run:

authorize("github")
metrics <- get_github_metrics(repo = "fhdsl/metricminer")

summary_metrics <- get_github_repo_summary(repo = "fhdsl/metricminer")
timecourse_metrics <- get_github_repo_timecourse(repo = "fhdsl/metricminer")

## End(Not run)
Description
This is a wrapper for get_github_metrics that has ‘time_course = FALSE’ so that summary metrics are collected
This is a function to get the information about a repository

Usage
get_github_repo_summary(
  repo,
  token = NULL,
  count = "all",
  data_format = "dataframe"
)

Arguments
repo The repository name. So for ‘https://github.com/fhdsl/metricminer’, it would be ‘fhdsl/metricminer’

token You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")’ function

count How many items would you like to receive? Put "all" to retrieve all records.

data_format Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".

Value
GitHub repository summary metrics

Examples
## Not run:
authorize("github")

summary_metrics <- get_github_repo_summary(repo = "fhdsl/metricminer")

## End(Not run)
get_github_repo_timecourse

Collect repository timecourse metrics

Description

This is a wrapper for `get_github_metrics` that has `time_course = TRUE` so that timecourse metrics are collected

This is a function to get the information about a repository

Usage

```r
get_github_repo_timecourse(
  repo,
  token = NULL,
  count = "all",
  data_format = "dataframe"
)
```

Arguments

- **repo**
  - The repository name. So for `https://github.com/fhdsl/metricminer`, it would be `fhdsl/metricminer`
- **token**
  - You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the `authorize("github")` function
- **count**
  - How many items would you like to receive? Put "all" to retrieve all records.
- **data_format**
  - Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".

Value

GitHub repository timecourse metrics for views and clones

Examples

```r
## Not run:
authorize("github")

timecourse_metrics <- get_github_repo_timecourse(repo = "fhdsl/metricminer")

## End(Not run)
```
get_github_user

Get the GitHub User’s info

Description
This is a function to get the GitHub user’s info.

Usage
get_github_user(token = NULL)

Arguments
token
You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")’ function.

Value
Information regarding a Github account.

Examples
## Not run:
authorize("github")
get_github_user()

## End(Not run)

get_google_form

Get Google Forms

Description
This is a function to get Google Form info and responses from the API.

Usage
get_google_form(form_id, token = NULL, dataformat = "dataframe")

Arguments
form_id
The form ID we need to get.
token
credentials for access to Google using OAuth. ‘authorize("google")’
dataformat
What format would you like the data? Options are "raw" or "dataframe". "dataframe" is the default.
get_multiple_forms

Value
This returns a list of the form info and responses to the google form. Default is to make this a list of nicely formatted dataframes.

Examples
## Not run:
authorize("google")
form_info <- get_google_form("https://docs.google.com/forms/d/1Neyj7wwNpn8wC7NzQND8kQ30cnbbETSpT0lKhX7uaQY/edit")

### OR You can give it a direct form id
form_info <- get_google_form("1Neyj7wwNpn8wC7NzQND8kQ30cnbbETSpT0lKhX7uaQY")

## End(Not run)

get_multiple_forms

Get multiple Google forms

Description
This is a wrapper function for returning google form info and responses for multiple forms at once

Usage
get_multiple_forms(form_ids = NULL, token = NULL)

Arguments
form_ids a vector of form ids you'd like to retrieve information for
token credentials for access to Google using OAuth. 'authorize("google")'

Value
This returns a list of API information for google forms

Examples
## Not run:
authorize("google")
form_list <- googledrive::drive_find(
  shared_drive = googledrive::as_id("0AJb5Zemj0AAkUK9PVA"),
  type = "form"
)
get_multiple_repos_metrics

Retrieve metrics for a list of repos

Description
This is a function to get metrics for a list of repos. You can provide an owner and attempt retrieve all repositories from a particular organization, or you can provide a character vector of repositories like "

Usage
get_multiple_repos_metrics(
  repo_names = NULL,
  token = NULL,
  data_format = "dataframe",
  time_course = FALSE
)

Arguments

repo_names a character vector of repositories you’d like to collect metrics from.
token You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function
data_format Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".time_course Should the time course data be collected or only the summary metrics?

Value
Information regarding a Github account

Examples

## Not run:
authorize("github")

repo_names <- c("fhdsl/metricminer", "jhudsl/OTTR_Template")
some_repos_metrics <- get_multiple_repos_metrics(repo_names = repo_names)
some_repos_metrics <- get_multiple_repos_metrics(repo_names = repo_names, time_course = TRUE)
get_org_repo_list

Retrieve list of repositories for an organization

Description

This is a function to get the information about a repository

Usage

get_org_repo_list(
  owner,
  count = "all",
  data_format = "dataframe",
  token = NULL
)

Arguments

owner        The owner of the repository. So for ‘https://github.com/fhdsl/metricminer’, it would be ‘fhdsl’
count       The number of responses that should be returned. Default is 20 or you can say "all" to retrieve all.
data_format  Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".
token        You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")’ function

Value

a list of repositories that an organization has

Examples

## Not run:
authorize("github")
get_org_repo_list(owner = "fhdsl")

## End(Not run)
get_question_metadata  

**Google Form handling functions**

**Description**

This is a function to get metadata about a Google Form. It is used by the `get_google_form()` function if `dataformat = "dataframe"`

**Usage**

```r
get_question_metadata(form_info)
```

**Arguments**

- `form_info`  
  The return form_info list that is extracted in `get_google_form()`

**Value**

This returns metadata from a google form

---

get_slido_files  

**Get Slido Files**

**Description**

This is a function to get slido response output files. The slido files must be saved as googlesheets and cannot be xlsx.

**Usage**

```r
get_slido_files(
  drive_id,  
  token = NULL,  
  recursive = TRUE,  
  keep_duplicates = FALSE
)
```

**Arguments**

- `drive_id`  
  a URL or drive id that has the slido response output files you are looking to get (will recursively search for files by default).
- `token`  
  credentials for access to Google using OAuth. `authorize("google")`
- `recursive`  
  Should slido files be looked for recursively in this folder? default is TRUE.
- `keep_duplicates`  
  By default we won’t keep duplicated files if a two files have the same name. But if you set this to true, duplicates will be returned.
Value

A list of the slido files and their content in a Googledrive location.

Examples

```r
# Not run:
drive_id <- "https://drive.google.com/drive/folders/0AJb5Zemj0AAkUk9PVA"
drive_id <- "https://drive.google.com/drive/u/0/folders/1XWXHHyj32Uw_UyaUIjrp6S--HhNw-71"
slido_data <- get_slido_files(drive_id)
```

## get_timestamp_repo_metrics

### Get timestamp repository metrics

Description

Get timestamp repository metrics

Usage

```r
get_timestamp_repo_metrics(results, column)
```

Arguments

- `results` An API result from GitHub typically the views or clones for a repo
- `column` name of the column being extracted. Typically "views" or "clones"

Value

Extracted timestamp metrics from the API response

### get_user_repo_list

Retrieve list of repositories for an organization

Description

This is a function to get the information about a repository
get_youtube_channel_stats

Usage

get_user_repo_list(
    owner,
    count = "all",
    data_format = "dataframe",
    token = NULL
)

Arguments

owner The owner of the repository. So for ‘https://github.com/fhdsl/metricminer’, it would be ‘fhdsl’

count The number of responses that should be returned. Default is 20 or you can say "all" to retrieve all.

data_format Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".

token You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")’ function

Value

a list of repositories that an organization has

Examples

## Not run:

authorize("github")
get_user_repo_list(owner = "metricminer")

## End(Not run)

get_youtube_channel_stats

Get Youtube channel stats

Description

This is a function to retrieve statistics for a Youtube channel

Usage

google_channel_stats(channel_id, token = NULL, dataformat = "dataframe")
get_youtube_video_stats

Arguments

- **channel_id**: ID of the Youtube channel to retrieve stats from.
- **token**: OAuth token from Google login.
- **dataformat**: How would you like the data returned to you? Default is a "dataframe" but if you’d like to see the original API list result, put "raw".

Value

A data frame of the channel stats from a Youtube channel.

Examples

```r
# Not run:
authorize("google")
youtube_channel_stats <- get_youtube_channel_stats("UCr73I9ZEPbn-3_1CBM57gQ")
# End(Not run)
```

get_youtube_video_stats

*Get Youtube video stats*

Description

This is a function to get a statistics on a Youtube video

Usage

```r
google_video_stats(\text{video_id}, \text{token} = \text{NULL}, \text{dataformat} = \"dataframe\")
```

Arguments

- **video_id**: ID of the Youtube video to retrieve stats from.
- **token**: OAuth token from Google login. https://www.youtube.com/watch?v=YkYnni-WuaQor just the "YkYnni-WuaQor" part that comes after the ‘v=’ bit.
- **dataformat**: How would you like the data returned to you? Default is a "dataframe" but if you’d like to see the original API list result, put "raw".

Value

A data frame of the Youtube video stats.
**gh_repo_wrapper**

### Examples

```r
## Not run:
authorize("google")
youtube_video_stats <- get_youtube_video_stats("YkYnni-WuaQ")

## End(Not run)
```

### gh_repo_wrapper

**Wrapper function for gh repository calls**

### Description

This is a function that wraps up gh calls for us.

### Usage

```r
gh_repo_wrapper(api_call, owner, repo, token = NULL, count = Inf)
```

#### Arguments

- `api_call`: an API call and endpoint. That has ‘owner’ and ‘user’.
- `owner`: The repository name. So for ‘https://github.com/fhdsl/metricminer’, it would be ‘fhdsl’
- `repo`: The repository name. So for ‘https://github.com/fhdsl/metricminer’, it would be ‘metricminer’
- `token`: You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")’ function
- `count`: How many items would you like to receive? Put "all" to retrieve all records.

#### Value

Metrics for a repository on GitHub

---

**key_encrypt_creds_path**

*Get file path to an key encryption RDS*

### Description

Get file path to an key encryption RDS

### Usage

```r
key_encrypt_creds_path()
```
list_calendly_events  

*Get Calendly Event Lists*

**Description**

This is a function to get a list of scheduled events from a Calendly user.

**Usage**

```r
list_calendly_events(token = NULL, user, count = 100)
```

**Arguments**

- `token`  
  You can provide the API key directly using this argument or this function will attempt to grab an API key that was stored using the `authorize("calendly")` function.

- `user`  
  You need to retrieve the Calendly user’s URI. You can do this by doing `user <- get_calendly_user()` and `user$resource$uri`

- `count`  
  The number of responses that should be returned. Default is 20 or you can say "all" to retrieve all.

**Value**

Calendly REST API response as a list

**Examples**

```r
## Not run:
authorize("calendly")
user <- get_calendly_user()
list_calendly_events(user = user$resource$uri)
## End(Not run)
```

list_example_data  

*Get list of example datasets*

**Description**

This is a function to retrieve a list of the example datasets included with metricminer

**Usage**

```r
list_example_data()
```
Value

A list of the example datasets available in this package

Examples

```r
## Not run:
list_example_data()

# Now you could use any of these example datasets that are printed out
get_example_data("calendly_events")

## End(Not run)
```

---

**request_ga**

*Handler for API requests from Google Analytics*

Description

This is a function that handles requests from Google Analytics

Usage

```r
request_ga(token, url, query = NULL, body_params = NULL, request_type)
```

Arguments

- **token** credentials for access to Google using OAuth. `authorize("google")`
- **url** The endpoint URL for the request
- **query** A list to be passed to query
- **body_params** The body parameters for the request
- **request_type** Is this a GET or a POST?

Value

An API response in the form of a list
request_google_forms  Get Google Forms

Description
This is a function to get the Calendly API user info

Usage
request_google_forms(
    token,
    url,
    body_params = NULL,
    query_params = NULL,
    return_request = TRUE
)

Arguments

    token          credentials for access to Google using OAuth. ‘authorize("google")’
    url            The endpoint URL for the request
    body_params    The body parameters for the request
    query_params   The body parameters for the request
    return_request Should a list of the request be returned as well?

Value
This function returns a list from a API response JSON file

supported_endpoints  Supported endpoints

Description
This is function stores endpoints and supported app names

Usage
supported_endpoints()
write_playlist_details

Write playlist details from YouTube

Description

Write playlist details from YouTube

Usage

write_playlist_details(playlist_id, token = NULL, outfile = NULL)

Arguments

playlist_id      string, playlist ID on YouTube
token            OAuth token from Google login.
outfile          string, a filename to which to write results in the 'resources' folder

Value

writes a file containing the dataframe of cleaned results

Examples

## Not run:
# Not run
write_playlist_details(playlist_id = shorts_playlist_id,
    outfile = "youtube_shorts_data.tsv")
write_playlist_details(
    playlist_id = "PL6aYJ_0zJ4uCABkMngSYjPo_3c-nUUmio",
    outfile = "youtube_shorts_data.tsv")

## End(Not run)

write_to_gsheet

Writes data to a Googlesheet

Description

This is a function to write metricminer data to a Googlesheet
Usage

write_to_gsheet(
    input,
    token = NULL,
    gsheet = NULL,
    overwrite = FALSE,
    append_rows = FALSE,
    sheet = 1,
    new_sheet = FALSE,
    ...
)

Arguments

input input data to write to a googlesheet
token OAuth token from Google login.
gsheet Optionally a googlesheet to write to
overwrite TRUE/FALSE overwrite if there is data at the destination
append_rows TRUE/FALSE should the data be appended to the data?
sheet Index or name of the worksheet you want to write to. Forwarded to googlesheets4::write_sheet or googlesheets4::append_sheet to indicate what sheet it should be written to.
new_sheet default is FALSE. But if it is anything else will be used as the name for a new worksheet that will be made and written to.
...
these parameters are sent to googlesheets4::write_sheet.

Value

The googlesheet URL where the data has been written

Examples

## Not run:
authorize("github")
repo_list <- get_user_repo_list(owner = "metricminer")
gsheet <- paste0("https://docs.google.com/spreadsheets/d/",
    "166MV4_1pAlB3Hes2HbdZCpkMc8JTTJ3u3eJes0Wu7RK/edit#gid=0")
write_to_gsheet(repo_list)

datasheet <- write_to_gsheet(
    gsheet = gsheet,
    input = repo_list, append_rows = TRUE,
    sheet = 1)

datasheet <- write_to_gsheet(
    gsheet = gsheet,
    input = repo_list,
    new_sheet = "github_data")
write_to_gsheet

```r
## End(Not run)
```
Index

app_set_up, 3
auth_from_secret, 4
authorize, 3
calendly_get, 5
check_check, 6
clean_ga_metrics, 7
clean_repo_metrics, 7
default_creds_path, 8
delete_creds, 8
encrypt_creds_path, 9
eexample_data_folder, 9
extract_answers, 9
get_all_ga_metrics, 10
calendly_user, 11
eexample_data, 11
gt_ga_metadata, 12
gt_ga_properties, 13
gt_ga_stats, 13
gt_ga_user, 15
gt_github, 15
gt_github_metrics, 16, 17, 18
gt_github_repo_summary, 17
gt_github_repo_timecourse, 18
gt_github_user, 19
gt_google_form, 19
gt_multiple_forms, 20
gt_multiple_repos_metrics, 21
gt_org_repo_list, 22
gt_question_metadata, 23
gt_slido_files, 23
gt_timestamp_repo_metrics, 24
gt_user_repo_list, 24
gt_youtube_channel_stats, 25
gt_youtube_video_stats, 26
gt_repo_wrapper, 27
key_encrypt_creds_path, 27
list_calendly_events, 28
list_example_data, 28
oauth2.0_token, 3
request_ga, 29
request_google_forms, 30
supported_endpoints, 30
write_playlist_details, 31
write_to_gsheet, 31