Package ‘piwikproR’

June 8, 2021

**Type**  Package

**Title**  Access 'Piwik Pro' Website Statistics

**Date**  2021-06-07

**Version**  0.2.0

**Author**  Martin Stingl <martin.stingl@dfv.de>

**Maintainer**  Martin Stingl <martin.stingl@dfv.de>

**Description**  Run Queries against the API of 'Piwik Pro' <https://developers.piwik.pro/en/latest/custom_reports/http_api/http_api.html>. The result is a tibble.

**URL**  https://github.com/dfv-ms/piwikproR

**BugReports**  https://github.com/dfv-ms/piwikproR/issues

**License**  GPL (>= 3)

**Encoding**  UTF-8

**Imports**  urltools, httr, rjson, tibble, purrr, lubridate, magrittr, dplyr, rlang, stringr, readr, digest, fs

**RoxygenNote**  7.1.1

**Suggests**  testthat (>= 3.0.0), knitr, rmarkdown

**Config/testthat/edition**  3

**VignetteBuilder**  knitr

**NeedsCompilation**  no

**Repository**  CRAN

**Date/Publication**  2021-06-08 06:50:02 UTC

\R topics documented:

- apply_types .................................................. 2
- build_filter .................................................. 2
- build_query .................................................. 4
- get_column_type ........................................... 5
- get_login_token ............................................ 5
apply_types

Convert column-type according to column_name

Description

Convert column-type according to column_name

Usage

apply_types(data)

Arguments

data tibble

Value

tibble

build_filter

Build filter from tribble

Description

Build filter from tribble

Usage

build_filter(filters, global_operator = "and")
Arguments

- `filters`: tribble with columns
  - column
  - operator
  - value

Possible values for operator:
- `(not)_contains`
- `(not)_icontains`
- `start_with`
- `ends_with`
- `(not)_matches`
- `eq`
- `neq`
- `(not)_empty`

- `global_operator`: "and" or "or"

Value

Filter json encoded to feed to `build_query`

Examples

```r
filters <- tibble::tribble(
  ~column, ~operator, ~value,
  "event_url", "matches", "Zamperoni",
  "event_url", "matches", "-1[34]"
)
# With optional transformation
filters <- tibble::tribble(
  ~column, ~operator, ~value, ~transformation,
  "event_url", "starts_with", "/medien", "to_path",
  "event_url", "matches", "-1[34]", NULL
)
filters <- build_filter(filters, "and")
# If values of two or more different types are used use lists
filters <- tibble::tribble(
  ~column, ~operator, ~value,
  "device_type", "eq", list(0),
  "location_country_name", "eq", list("DE")
)
```
build_query
Build the query

Description
Build the query

Usage
build_query(
  date_from,
  date_to,
  website_id,
  columns,
  filters = NULL,
  metric_filters = NULL,
  offset = 0,
  max_lines = 0
)

Arguments
date_from Start date of query
date_to End date of query
website_id website_id from piwik
columns tibble containing columns and transformations (metrics and dimensions)
filters list containing filter, best built by build_filter
metric_filters list containing filter, best built by build_filter
offset offset
max_lines limit

Value
query as list

Examples
columns <- tibble::tribble(
  ~column, ~transformation,
  "event_url", "to_path",
  "event_url", "to_domain",
  "website_name", "",
  "timestamp", "to_hour_of_day",
  "timestamp", "to_hour_of_day",
  "page_views", ""
)
get_column_type

Description
Convert column-type according to column_name

Usage
get_column_type(column_name)

Arguments
column_name  string

Value
string suggested type of column

get_login_token

Description
Fetch login token

Usage
get_login_token(credentiaLs)

Arguments
credentials  List with fields client_id, client_secret and url

Value
List with login_token
### get_test_credentials

**Description**

Fills credentials out of ENV into a list

**Usage**

```python
get_test_credentials()
```

**Value**

- list

---

### is_column_a_metric

**Description**

Checks if column_name indicates numeric values Uses https://developers.piwik.pro/en/latest/custom_reports/columns.html

**Usage**

```python
is_column_a_metric(column_name)
```

**Arguments**

- column_name: string

**Value**

- boolean
**MAX_LINES_PER_REQUEST**

*Maximum number of line requested*

**Description**

Maximum number of line requested

**Usage**

`MAX_LINES_PER_REQUEST()`

**Value**

`int`

---

**release_questions**

*Ask package specific question during release-process*

**Description**

Ask package specific question during release-process

**Usage**

`release_questions()`

**Value**

`vector of strings`

---

**send_query**

*Send the query and receive the result*

**Description**

Send the query and receive the result
Usage

send_query(
    query,
    token,
    use_csv = TRUE,
    fetch_by_day = FALSE,
    api = "query",
    caching = FALSE,
    caching_dir = "cache",
    convert_types = TRUE
)

Arguments

query       list generated by build_query
token       login token
use_csv     logical to choose whether to fetch data via extra csv-request
fetch_by_day logical fetch data day by day
api         API endpoint ("query", "sessions", "events")
caching     logical Set TRUE to enable caching
caching_dir character Set directory for saving caching data, default cache
convert_types logical guess type of columns and set them

Value

result as tibble

send_query_single Send the query and receive the result

Description

Send the query and receive the result

Usage

send_query_single(query, token, use_csv, api, caching, caching_dir)

Arguments

query       list generated by build_query()
token       login token
use_csv     logical to choose whether to fetch data via extra csv-request
api         API endpoint (query, sessions, events)
caching     logical Set TRUE to enable caching
caching_dir character Set directory for saving caching data
send_query_single

Value

result as list with values data and meta
Index

apply_types, 2
build_filter, 2, 4
build_query, 3, 4, 8
get_column_type, 5
get_login_token, 5
get_test_credentials, 6
is_column_a_metric, 6
MAX_LINES_PER_REQUEST, 7
release_questions, 7
send_query, 7
send_query_single, 8