Package 'zdeskR'

July 1, 2025

Title Connect to Your 'Zendesk' Data

Version 0.6.0

Description Facilitates making a connection to the 'Zendesk' API and executing various queries. You can use it to get ticket, ticket metrics, and user data. The 'Zendesk' documentation is available at <https://developer.zendesk.com/rest_api/docs/support/introduction>. This package is not supported by 'Zendesk' (owner of the software).

URL https://github.com/chrisumphlett/zdeskR

BugReports https://github.com/chrisumphlett/zdeskR/issues

License CC0

Encoding UTF-8

Imports dplyr (>= 1.0.0), magrittr (>= 1.5), jsonlite (>= 1.6.1), purrr (>= 0.3.3), httr (>= 1.4.1), tidyr (>= 1.0.0), plyr (>= 1.8.6), tidyselect (>= 1.2.0)

RoxygenNote 7.3.2

NeedsCompilation no

Author Chris Umphlett [aut, cre], Avinash Panigrahi [aut]

Maintainer Chris Umphlett <christopher.umphlett@gmail.com>

Repository CRAN

Date/Publication 2025-07-01 17:10:02 UTC

Contents

`get_all_ticket_metrics` *Get Metrics on All Zendesk Tickets*

Description

This function takes your Email Id, authentication token, and sub-domain and parses all the tickets and its corresponding metrics in a list. Since each iteration only returns 100 tickets at a time you must run the loop until the "has_more" parameter is equal to FALSE.

Usage

```
get_all_ticket_metrics(email_id, token, subdomain)
```

Arguments

`email_id`	Zendesk Email Id (username).
`token`	Zendesk API token.
`subdomain`	Your organization's Zendesk sub-domain.

Details

Its not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

Value

Data Frame with metrics for all tickets

References

https://developer.zendesk.com/rest_api/docs/support/ticket_metrics

Examples

```
## Not run:
ticket_metrics <- get_all_ticket_metrics(email_id, token, subdomain)

## End(Not run)
```

`get_custom_fields` *Returns the system and all the custom fields defined by your organization's zendesk administrator*

Description

It takes your Email Id, authentication token, sub-domain as parameters and gets the system and all the custom fields available for a zendesk ticket.

Usage

```
get_custom_fields(email_id, token, subdomain)
```

Arguments

`email_id`	Zendesk Email Id (username).
`token`	Zendesk API token.
`subdomain`	Your organization's Zendesk sub-domain.

Details

It's not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

Value

A data frame containing all ticket fields

References

https://developer.zendesk.com/rest_api/docs/support/ticket_fields

Examples

```
## Not run:
fields <- get_custom_fields(email_id, token, subdomain)

## End(Not run)
```

`get_satisfaction_ratings` *Get Ticket Satisfaction Ratings*

Description

This function takes your Email Id, authentication token, sub-domain and start time as parameters and gets all the satisfaction ratings for tickets which have been received on or after the start time parameter. By default each page returns 100 unique tickets and a next page url value which stores a pointer to the next page (by updating the start time parameter). After getting the first page this function will then loop through all subsequent pages until there are none left.

Usage

```
get_satisfaction_ratings(
  email_id,
  token,
  subdomain,
  start_time,
  rating_type = "received"
)
```

Arguments

`email_id`	Zendesk Email Id (username).
`token`	Zendesk API token.
`subdomain`	Your organization's Zendesk sub-domain.
`start_time`	String with a date or datetime to get all tickets modified after that date.
`rating_type`	String that specifies whether you want to see all received ratings, offered ratings, or unoffered ratings.

Details

It's not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

The start time parameter should be in 'UTC' format as Zendesk uses the 'UTC' time zone when retrieving tickets after the start time. For example, the US Eastern Time Zone is currently four hours behind UTC. If one wanted to get tickets starting on August 1 at 12 am, you would need to enter "2020-08-01 04:00:00". The user must do proper adjustment to accommodate the time zone difference, if desired.

rating_type allows you to get the satisfaction ratings, or, to see the tickets where a user was offered the opportunity to respond and did not, or to see the tickets where a user was not offered the survey. By default it will do received.

Value

a Data Frame containing all tickets, satisfaction ratings, and the comments.

References

https://developer.zendesk.com/api-reference/ticketing/ticket-management/satisfaction_ratings/

Examples

```
## Not run:
ratings <- get_satisfaction_ratings(email_id, token, subdomain,
  start_time = "2021-01-31 00:00:00", rating_type = "received")
)

## End(Not run)
```

---

get_tickets *Get Zendesk Tickets*

Description

This function takes your Email Id, authentication token, sub-domain and start time as parameters and gets all the tickets which have been updated on or after the start time parameter. By default each page returns 1000 unique tickets and an "after_cursor" value which stores a pointer to the next page. After getting the first page it uses the pointer to fetch the subsequent pages.

Usage

```
get_tickets(email_id, token, subdomain, start_time, remove_cols = NULL)
```

Arguments

email_id	Zendesk Email Id (username).
token	Zendesk API token.
subdomain	Your organization's Zendesk sub-domain.
start_time	String with a date or datetime to get all tickets modified after that date.
remove_cols	Vector of column names to remove from the results.

Details

The start time parameter should be in 'UTC' format as Zendesk uses the 'UTC' time zone when retrieving tickets after the start time. For example, the US Eastern Time Zone is currently four hours behind UTC. If one wanted to get tickets starting on August 1 at 12 am, you would need to enter "2020-08-01 04:00:00". The user must do proper adjustment to accommodate the time zone difference, if desired. A date can be provided, it will retrieve results as of 12 am in the UTC time zone.

Start and end times can be entered with or without the time component. End time cannot be in the future, but should work for values up to one minute prior to the current time.

It's not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

The remove_cols parameter allows the removal of custom fields causing errors. Errors occurred when a field was sometimes blank and assigned a logical type and then appended to non-blank, non-logical inside of purrr::map_dfr. See issue #1 on GH.

Value

a Data Frame containing all tickets after the start time.

References

https://developer.zendesk.com/rest_api/docs/support/incremental_export#start_time

Examples

```
## Not run:
all_tickets <- get_tickets(email_id, token, subdomain,
  start_time = "2021-01-31 00:00:00"
)

## End(Not run)
```

---

get_tickets_comments *Get tickets comments/replies*

Description

This function takes your email ID, authentication token, sub-domain, and specific ticket ID to fetch all comments/replies to this wanted ticket.

Usage

```
get_tickets_comments(
  email_id,
  token,
  subdomain,
  ticket_id,
  add_cols = NULL,
  metadata = FALSE
)
```

Arguments

email_id	Zendesk Email ID (username).
token	Zendesk API token.
subdomain	Your organization's Zendesk sub-domain.
ticket_id	The ticket ID number. A numeric value.
add_cols	Vector of column names to select in addition to the default.
metadata	Logical value (TRUE or FALSE). If TRUE, metadata columns will be included. This is set to FALSE by default.

Details

By default only these columns are returned: "id", "type", "author_id", "body", "created_at", "have_attachments". You can add other variables using the 'add_cols' parameter. The variables that can be inserted are described in the Zendesk API documentation: https://developer.zendesk.com/api-reference/ ticketing/tickets/ticket_comments/.

The meaning of the default columns included are described in the previous link, except "have-attachments" which is a boolean field that will be "Yes" if the comment has an attachment or "No" if it does not. The attachment itself cannot be returned.

If you request the 'metadata' sensitive data (location, lat, long, IP address, etc.) will be included. This data should be handled with care and only stored and used per your organization's policies and applicable privacy regulations.

Value

a Data Frame containing all comments/replies for a single ticket.

References

https://developer.zendesk.com/api-reference/ticketing/tickets/ticket_comments/

Examples

```
## Not run:
## Extracting comments with default columns and without sensitive data
comments_ticket_id <- get_tickets_comments(email_id, token, subdomain,
ticket_id, add_cols = NULL, metadata = FALSE)

## Extracting comments with additional columns and sensitive data
comments_ticket_id <- get_tickets_comments(email_id, token, subdomain,
ticket_id, add_cols = c("html_body", "attachments"), metadata = TRUE)

## End(Not run)
```

`get_ticket_audits` *Get Zendesk Ticket Audits*

Description

This function takes your Email Id, authentication token, sub-domain and ticket id as parameters and gets the first 100 audits. Pagination to get additional audits was not set up in the first version of this function.

Usage

```
get_ticket_audits(email_id, token, subdomain, ticket_id)
```

Arguments

`email_id`	Zendesk Email Id (username).
`token`	Zendesk API token.
`subdomain`	Your organization's Zendesk sub-domain.
`ticket_id`	Integer with Zendesk ticket id.

Details

It's not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

Value

a Data Frame containing first 100 audits for the ticket, with the events as a nested data frame in each row.

References

https://developer.zendesk.com/api-reference/ticketing/tickets/ticket_audits/

Examples

```
## Not run:
all_tickets <- get_ticket_audits(email_id, token, subdomain,
  ticket_id = 123456
)

## End(Not run)
```

`get_users` *Returns All Available Zendesk Users.*

Description

It takes your Email Id, authentication token, sub-domain and parse all the users in a list. It iterates through all the pages returning only 100 users per page until the "next_page" parameter becomes null indicating there are no more pages to fetch.

Usage

```
get_users(email_id, token, subdomain, start_time, user_role = "all")
```

Arguments

`email_id`	Zendesk Email Id (username).
`token`	Zendesk API token.
`subdomain`	Your organization's Zendesk sub-domain.
`start_time`	String with a date or datetime to get all tickets modified after that date.
`user_role`	User role, one of "all", "end-user", "agent", or "admin".

Details

It's not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

The start_page parameter is useful if you have many users. Each page contains 100 users. Zendesk does not have an incremental method for pulling users by date but after you retrieve all of your users once, you can then increment your start page to something that will limit the number of users you are re-pulling each time.

If you are pulling partial lists of users be aware that you will not get updates on older users. You will only get recently created users, not modified/deleted users and their modified data nor updated last login dates.

Value

Data Frame with user details

References

https://developer.zendesk.com/rest_api/docs/support/users

Examples

```
## Not run:
users <- get_users(email_id, token, subdomain, start_time = "2025-01-01 12:00:00")

## End(Not run)
```

`ticket_search` *Returns ticket data for a provided Zendesk search query*

Description

It takes your Email Id, authentication token, sub-domain and search query and returns all the tickets that meet the search criteria. 100 tickets are returned at a time. If your search query has many results, the function may run for a long time as it goes through each page of results.

Usage

```
ticket_search(email_id, token, subdomain, query)
```

Arguments

`email_id`	Zendesk Email Id (username).
`token`	Zendesk API token.
`subdomain`	Your organization's Zendesk sub-domain.
`query`	Zendesk search query to execute.

Details

It's not a good practice to write down these authentication parameters in your code. There are various methods and packages available that are more secure; this package doesn't require you to use any one in particular.

Value

Data Frame with user details

References

https://developer.zendesk.com/api-reference/ticketing/ticket-management/search/#list-search-results

Examples

```
## Not run:
search_results <- ticket_search(email_id, token, subdomain,
query = "query=satisfaction:goodwithcomment updated>24hours")

## End(Not run)
```

Index