

Package ‘smartsheetr’

July 23, 2025

Title Access and Write 'Smartsheet' Data using the 'Smartsheet' API
2.0

Version 0.1.0

Description Interact with the 'Smartsheet' platform through the 'Smartsheet' API 2.0. <<https://smartsheet.redoc.ly/>>. API is an acronym for application programming interface; the 'Smartsheet' API allows users to interact with 'Smartsheet' sheets directly within R.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 4.1.0)

Imports dplyr, httr, jsonlite, memoise, purrr, rlang, tibble, tidyr

Suggests devtools, knitr, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Cole Johanson [aut, cre, cph]

Maintainer Cole Johanson <coldenjohanson@gmail.com>

Repository CRAN

Date/Publication 2023-10-28 15:00:12 UTC

Contents

random_sheet_name	2
ss_add_columns	3
ss_add_rows	3
ss_api	4
ss_cols_to_dataframe	5
ss_column_ids	5
ss_column_type	6
ss_column_type_to_class	6
ss_delete_columns	7

ss_delete_rows	7
ss_delete_sheet	8
ss_get	9
ss_list_sheets	9
ss_list_sheet_shares	10
ss_list_users	11
ss_read_sheet	11
ss_rename_columns	12
ss_replace_sheet	13
ss_resp_data_to_dataframe	13
ss_row_ids	14
ss_sheetid	14
ss_sheet_share	15
ss_write_sheet	16
ss_write_sheet_columns	17
unlist_and_replace_null	18
validate_ss_id	18
Index	19

random_sheet_name	<i>Get a random sheet name</i>
-------------------	--------------------------------

Description

Randomly selects letters for a Smartsheet sheet name

Usage

```
random_sheet_name(n = 10)
```

Arguments

n The number of characters to generate

Value

A character vector

Examples

```
random_sheet_name()
```

ss_add_columns	<i>Add columns to an existing sheet</i>
----------------	---

Description

Add columns to an existing sheet

Usage

```
ss_add_columns(ss_id, data, index = 0)
```

Arguments

- ss_id The sheetId, permalink, or name of the Smartsheet sheet to read
- data A data frame of columns to be added
- index The index location where the columns should be added

Value

A ss_addcolumns_resp object

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-",random_sheet_name())))
ss_add_columns(ss_id, data.frame("FK"=character()), index=1)
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_add_rows	<i>Add rows to a sheet.</i>
-------------	-----------------------------

Description

Add rows to a sheet.

Usage

```
ss_add_rows(ss_id, data, column_ids = NULL)
```

Arguments

ss_id	The sheetId, permalink, or name of the Smartsheet sheet to read
data	A data frame of rows to be added
column_ids	A vector of the columnIds of the smartsheets sheetId. If NULL, this will be obtained.

Value

A ss_addrows_resp object

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name())))
ss_add_rows(ss_id, data.frame("PK"="1"))
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_api

The workhorse function that performs each call to the Smartsheet API

Description

The workhorse function that performs each call to the Smartsheet API

Usage

```
ss_api(FUN, ...)
```

Arguments

FUN	An http verb function, typically from the <code>httr</code> package
...	Further parameters passed to the http verb function

ss_cols_to_dataframe	<i>Helper function to take columns data and create a data frame.</i>
----------------------	--

Description

Helper function to take columns data and create a data frame.

Usage

```
ss_cols_to_dataframe(ss_cols_data)
```

Arguments

ss_cols_data	A data frame
--------------	--------------

ss_column_ids	<i>List column ids for a given sheet</i>
---------------	--

Description

Returns a vector of the Smartsheet internal column ids for a given sheet

Usage

```
ss_column_ids(ss_id)
```

Arguments

ss_id	The sheetId, permalink, or name of the Smartsheet sheet to read
-------	---

Value

A numeric vector

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name()))))
col_names = colnames(ss_read_sheet(ss_id))
col_ids = ss_column_ids(ss_id)
setNames(col_ids, col_names)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_column_type	<i>Return the Smartsheet Column Type that aligns with the R class</i>
----------------	---

Description

Return the Smartsheet Column Type that aligns with the R class

Usage

```
ss_column_type(r_class)
```

Arguments

r_class A character vector (returned from a call to `base::class()`)

Details

See <https://smartsheet.redoc.ly/tag/columnsRelated/#section/Column-Types>

Value

A character vector

ss_column_type_to_class	<i>Return an empty vector of the correct class from the smartsheet Column Type</i>
-------------------------	--

Description

The opposite of [ss_column_type](#)

Usage

```
ss_column_type_to_class(ss_column_type)
```

Arguments

ss_column_type A character vector

Details

See <https://smartsheet.redoc.ly/tag/columnsRelated/#section/Column-Types>

Value

A character vector

ss_delete_columns	Delete non-primary columns from a given sheet.
-------------------	--

Description

The primary column(s) cannot be deleted.

Usage

```
ss_delete_columns(ss_id, column_ids = NULL)
```

Arguments

ss_id	The sheetId, permalink, or name of the Smartsheet sheet to read
column_ids	A vector of the smartsheet rowIds, or NULL to delete all non-primary columns

Value

A list of ss_resp objects

Examples

```
## Not run:
df = data.frame(PK=c(1,2), FK=c("a","b"))
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-",random_sheet_name()), data=df))
col_ids = ss_column_ids(ss_id)
ss_delete_columns(ss_id, col_ids[2])
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_delete_rows	Delete rows from a given sheet
----------------	--------------------------------

Description

Delete rows from a given sheet

Usage

```
ss_delete_rows(ss_id, row_ids = NULL)
```

Arguments

ss_id The sheetId, permalink, or name of the Smartsheet sheet to read
 row_ids A vector of the smartsheet rowIds, or NULL to delete all

Value

A list of ss_resp objects

Examples

```
## Not run:
df = data.frame(PK=c(1,2), FK=c("a","b"))
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-",random_sheet_name()), data=df))
row_ids = ss_row_ids(ss_id)
ss_delete_rows(ss_id, row_ids[2])
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_delete_sheet	<i>Delete a smartsheet</i>
-----------------	----------------------------

Description

Delete a smartsheet

Usage

```
ss_delete_sheet(ss_id)
```

Arguments

ss_id The sheetId, permalink, or name of the Smartsheet sheet to read

Value

A ss_resp object

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-",random_sheet_name()))))
ss_read_sheet(ss_id)
ss_delete_sheet(ss_id)

## End(Not run)
```

`ss_get`*Execute curl commands for the Smartsheet API*

Description

`ss_get()` wraps the `httr::GET()` function `ss_post()` wraps the `httr::POST()` function `ss_put()` wraps the `httr::PUT()` function `ss_delete()` wraps the `httr::DELETE()` function

Usage

```
ss_get(path, ...)
```

```
ss_post(path, body, ...)
```

```
ss_delete(path, ...)
```

```
ss_put(path, ...)
```

Arguments

<code>path</code>	A character vector to add to the API url. See (https://smartsheet.redoc.ly/#section/Introduction) for more information.
<code>...</code>	Further arguments passed to ss_api
<code>body</code>	A list of objects

Details

Note that the environment variable `SMARTSHEET_API_TOKEN` should be defined in order to run this or any other `smarsheetr` functions.

Value

An `httr::response` object

`ss_list_sheets`*Get a data frame describing the smartsheets available*

Description

Get a data frame describing the smartsheets available

Usage

```
ss_list_sheets()
```

Details

Note that the environment variable SMARTSHEET_API_TOKEN should be defined in order to run this or any other smarsheetr functions.

Value

A dataframe

Examples

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

ss_list_sheet_shares *List share data for a given sheet*

Description

List share data for a given sheet

Usage

```
ss_list_sheet_shares(ss_id)
```

Arguments

ss_id The sheetId, permalink, or name of the Smartsheet sheet to read

Value

A dataframe

Examples

```
## Not run:  
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name())))  
ss_list_sheet_shares(ss_id)  
# clean up  
ss_delete_sheet(ss_id)  
  
## End(Not run)
```

ss_list_users	<i>List smartsheet users</i>
---------------	------------------------------

Description

List smartsheet users

Usage

```
ss_list_users()
```

Value

A dataframe

Examples

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

ss_read_sheet	<i>Reads a Smartsheet sheet into an R data frame</i>
---------------	--

Description

Reads a Smartsheet sheet into an R data frame

Usage

```
ss_read_sheet(ss_id)
```

Arguments

ss_id	The sheetId, permalink, or name of the Smartsheet sheet to read
-------	---

Value

A tibble::tbl_df object

Examples

```
## Not run:
df = mtcars
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name()), data=df))
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_rename_columns	<i>Rename columns</i>
-------------------	-----------------------

Description

Rename a set of columns. One of the following must be true:

- column_names is not NULL
- column_locs is not NULL, or
- new_names is the same length as the number of columns of the ss_id sheet

Usage

```
ss_rename_columns(ss_id, new_names, column_names = NULL, column_locs = NULL)
```

Arguments

ss_id	The sheetId, permalink, or name of the Smartsheet sheet to read
new_names	A character vector of new names for the chosen columns
column_names	A vector of names of columns within the sheet to be replaced
column_locs	A vector of locations of columns within the sheet to be replaced

Value

A list of ss_resp objects

Examples

```
## Not run:
df = data.frame("PK"=character(), "temp"=character())
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name()), data=df))
ss_rename_columns(ss_id, new_names="FK", column_names="temp")
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_replace_sheet	<i>Replace the contents of a sheet with a new data frame</i>
------------------	--

Description

Replace the contents of a sheet with a new data frame

Usage

```
ss_replace_sheet(ss_id, data)
```

Arguments

ss_id	The sheetId, permalink, or name of the Smartsheet sheet to read
data	A data frame

Value

A named list of ss_resp objects

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name())))
ss_replace_sheet(ss_id, data=mtcars)
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_resp_data_to_dataframe	<i>Helper to rbind lists in a list into a data frame</i>
---------------------------	--

Description

Helper to rbind lists in a list into a data frame

Usage

```
ss_resp_data_to_dataframe(resp_data)
```

Arguments

resp_data	A list of lists
-----------	-----------------

ss_row_ids	<i>List row ids for a given sheet</i>
------------	---------------------------------------

Description

Returns a vector of the Smartsheet internal row ids for a given sheet

Usage

```
ss_row_ids(ss_id)
```

Arguments

ss_id The sheetId, permalink, or name of the Smartsheet sheet to read

Value

A numeric vector

Examples

```
## Not run:
df = data.frame(PK=c(1,2), FK=c("a","b"))
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-",random_sheet_name()), data=df))
ss_row_ids(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_sheetid	<i>Get a smartsheet sheetId from a response</i>
------------	---

Description

Get a smartsheet sheetId from a response

Usage

```
ss_sheetid(resp)
```

Arguments

resp An ss_resp object

Value

A numeric sheetId

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name()))))

## End(Not run)
```

ss_sheet_share	<i>Share a sheet with a user</i>
----------------	----------------------------------

Description

Share a sheet with a user

Usage

```
ss_sheet_share(
  ss_id,
  email,
  access_level = c("VIEWER", "EDITOR", "COMMENTER", "EDITOR_SHARE", "OWNER", "ADMIN")
)
```

Arguments

ss_id	The sheetId (or permalink) of the table
email	The email address of the user to share to, i.e. a value in <code>ss_list_users()\$email</code>
access_level	A character object. See https://smartsheet.redoc.ly/#section/Security/Access-Levels

Value

An `ss_resp` object

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name()))))
users = ss_list_users()
user = users[1, 'email']
ss_sheet_share(ss_id, user)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

ss_write_sheet

Create a sheet

Description

Creating a sheet requires either a template or a set of columns (see <https://smartsheet.redoc.ly/tag/sheets#operation/create-sheet-in-sheets-folder>). This function only allows for the columns option.

Usage

```
ss_write_sheet(
  sheet_name,
  data = data.frame(PK = character()),
  use_rownames = FALSE
)
```

Arguments

sheet_name	A character vector
data	A data frame
use_rownames	Logical; whether to use the rownames as the Primary Column

Details

The **Smartsheet API 2.0** uses two calls for creating a sheet with data. The first is a call to create a sheet and populate the columns (analogous to [ss_write_sheet_columns](#)). The second is to add rows (analogous to [ss_add_rows](#)). [ss_write_sheet](#) accomplishes both of these steps.

Value

A smartsheetr response object

Examples

```
## Not run:
ss_id = ss_sheetid(ss_write_sheet(paste0("smartsheetr-example-", random_sheet_name()), data=mtcars))
ss_read_sheet(ss_id)
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

`ss_write_sheet_columns`*Write the initial columns for the a sheet*

Description

Write the initial columns for the a sheet

Usage

```
ss_write_sheet_columns(sheet_name, data = data.frame(PK = character()))
```

Arguments

<code>sheet_name</code>	A character vector
<code>data</code>	A data frame of columns to be added

Details

The **Smartsheet API 2.0** uses two calls for creating a sheet with data. The first is a call to create a sheet and populate the columns (analogous to [ss_write_sheet_columns](#)). The second is to add rows (analogous to [ss_add_rows](#)). [ss_write_sheet](#) accomplishes both of these steps.

Value

A `ss_createsheet_resp` object

Examples

```
## Not run:
temp_sheet_name = paste0("smartsheetr-example-",random_sheet_name())
ss_id = ss_sheetid(ss_write_sheet_columns(temp_sheet_name, data=mtcars))
ss_read_sheet(ss_id) # No rows. Use ss_write_sheet() to write the full data frame
# clean up
ss_delete_sheet(ss_id)

## End(Not run)
```

`unlist_and_replace_null`

Helper function to replace NULL values with NA, and unlist, which is useful in converting nested lists to data frames

Description

Helper function to replace NULL values with NA, and unlist, which is useful in converting nested lists to data frames

Usage

```
unlist_and_replace_null(l)
```

Arguments

l	A list
---	--------

`validate_ss_id`

Validate or get the sheetID from a numeric/character vector

Description

This function validates a single ss_id is passed in and returns a smartsheets sheetId

Usage

```
validate_ss_id(ss_id)
```

Arguments

ss_id	A smartsheet sheet name, permalink, of sheetId
-------	--

Value

A smartsheets sheetId

Index

random_sheet_name, [2](#)

ss_add_columns, [3](#)
ss_add_rows, [3](#), [16](#), [17](#)
ss_api, [4](#), [9](#)
ss_cols_to_dataframe, [5](#)
ss_column_ids, [5](#)
ss_column_type, [6](#), [6](#)
ss_column_type_to_class, [6](#)
ss_delete(ss_get), [9](#)
ss_delete_columns, [7](#)
ss_delete_rows, [7](#)
ss_delete_sheet, [8](#)
ss_get, [9](#)
ss_list_sheet_shares, [10](#)
ss_list_sheets, [9](#)
ss_list_users, [11](#)
ss_post(ss_get), [9](#)
ss_put(ss_get), [9](#)
ss_read_sheet, [11](#)
ss_rename_columns, [12](#)
ss_replace_sheet, [13](#)
ss_resp_data_to_dataframe, [13](#)
ss_row_ids, [14](#)
ss_sheet_share, [15](#)
ss_sheetid, [14](#)
ss_write_sheet, [16](#), [16](#), [17](#)
ss_write_sheet_columns, [16](#), [17](#), [17](#)

unlist_and_replace_null, [18](#)

validate_ss_id, [18](#)