gspread Documentation

Release 3.0.1

Anton Burnashev

Oct 18, 2019

Contents

1 Main Interface **3**

2 Models **7**

3 Utils **15**
3.1 gspread.utils 15

4 Exceptions **17**

5 Indices and tables **19**

Python Module Index **21**

Index **23**

gspread is an asynchronous Python client library for the Google Sheets API based off of gspread by burnash.

- *Main Interface*
- *Models*
- *Utils*
  - *gspread.utils*
- *Exceptions*

CHAPTER 1

Main Interface

class aiospread.**Client**(*auth*, *session=None*)

An instance of this class communicates with Google API.

Parameters

- **auth** – An OAuth2 credential object. Credential objects are those created by the oauth2client library. https://github.com/google/oauth2client
- **session** – (optional) A session object capable of making HTTP requests while persisting some parameters across requests. Defaults to requests.Session.

```
>>> c = gspread.Client(auth=OAuthCredentialObject)
```

create(*title*)

This function is a *coroutine*.

Creates a new spreadsheet.

Parameters **title** – A title of a new spreadsheet.

Returns a `Spreadsheet` instance.

Note: In order to use this method, you need to add `https://www.googleapis.com/auth/drive` to your oAuth scope.

Example:

```
scope = [
    'https://spreadsheets.google.com/feeds',
    'https://www.googleapis.com/auth/drive'
]
```

Otherwise you will get an `Insufficient Permission` error when you try to create a new spreadsheet.

---

del_spreadsheet (*file_id*)
This function is a *coroutine*.

Deletes a spreadsheet.

Parameters **file_id** – a spreadsheet ID (aka file ID.)

import_csv (*file_id*, *data*)
This function is a *coroutine*.

Imports data into the first page of the spreadsheet.

Parameters

- **file_id** – The file ID of the sheet
- **data** – A CSV string of data.

insert_permission (*file_id*, *value*, *perm_type*, *role*, *notify=True*, *email_message=None*)
This function is a *coroutine*.

Creates a new permission for a file.

Parameters

- **file_id** – a spreadsheet ID (aka file ID.)
- **value** – user or group e-mail address, domain name or None for 'default' type.
- **perm_type** – the account type. Allowed values are: `user`, `group`, `domain`, `anyone`
- **role** – the primary role for this user. Allowed values are: `owner`, `writer`, `reader`
- **notify** – Whether to send an email to the target user/domain.
- **email_message** – an email message to be sent if notify=True.

Examples:

```
# Give write permissions to otto@example.com

gc.insert_permission(
    '0BmgG6nO_6dprnRRUWl1UFE',
    'otto@example.org',
    perm_type='user',
    role='writer'
)

# Make the spreadsheet publicly readable

gc.insert_permission(
    '0BmgG6nO_6dprnRRUWl1UFE',
    None,
    perm_type='anyone',
    role='reader'
)
```

list_permissions (*file_id*)
This function is a *coroutine*.

Retrieve a list of permissions for a file.

Parameters **file_id** – a spreadsheet ID (aka file ID.)

login()
This function is a *coroutine*.

Authorize client.

open(*title*)
This function is a *coroutine*.

Opens a spreadsheet.

Parameters **title** – A title of a spreadsheet.

Returns a `Spreadsheet` instance.

If there's more than one spreadsheet with same title the first one will be opened.

Raises **gspread.SpreadsheetNotFound** – if no spreadsheet with specified *title* is found.

```
>>> c = await gspread.authorize(credentials)
>>> await c.open('My fancy spreadsheet')
```

open_by_key(*key*)
This function is a *coroutine*.

Opens a spreadsheet specified by *key*.

Parameters **key** – A key of a spreadsheet as it appears in a URL in a browser.

Returns a `Spreadsheet` instance.

```
>>> c = await gspread.authorize(credentials)
>>> await c.open_by_key('0BmgG6nO_6dprdS1MN3d3MkdPa142WFRrdnRRUWl1UFE')
```

open_by_url(*url*)
This function is a *coroutine*.

Opens a spreadsheet specified by *url*.

Parameters **url** – URL of a spreadsheet as it appears in a browser.

Returns a `Spreadsheet` instance.

Raises **gspread.SpreadsheetNotFound** – if no spreadsheet with specified *url* is found.

```
>>> c = await gspread.authorize(credentials)
>>> await c.open_by_url('https://docs.google.com/spreadsheet/ccc?key=0Bm...FE&
↪hl')
```

openall(*title=None*)
This function is a *coroutine*.

Opens all available spreadsheets.

Parameters **title** – (optional) If specified can be used to filter spreadsheets by title.

Returns a list of `Spreadsheet` instances.

remove_permission(*file_id*, *permission_id*)
Deletes a permission from a file.

Parameters

- **file_id** – a spreadsheet ID (aka file ID.)
- **permission_id** – an ID for the permission.

CHAPTER 2

Models

The models represent common spreadsheet objects: `a spreadsheet`, `a worksheet` and `a cell`.

Note: The classes described below should not be instantiated by end-user. Their instances result from calling other objects' methods.

class aiospread.models.**Spreadsheet**(*client*, *properties*)

The class that represents a spreadsheet.

add_worksheet(*title*, *rows*, *cols*)

This function is a *coroutine*.

Adds a new worksheet to a spreadsheet.

Parameters

- **title** – A title of a new worksheet.
- **rows** – Number of rows.
- **cols** – Number of columns.

Returns a newly created `worksheets`.

del_worksheet(*worksheet*)

This function is a *coroutine*.

Deletes a worksheet from a spreadsheet.

Parameters **worksheet** – The worksheet to be deleted.

get_worksheet(*index*)

This function is a *coroutine*.

Returns a worksheet with specified *index*.

Parameters **index** – An index of a worksheet. Indexes start from zero.

Returns an instance of `gsperad.models.Worksheet` or *None* if the worksheet is not found.

Example. To get first worksheet of a spreadsheet:

```
>>> async def sheets():
...     sht = await client.open('My fancy spreadsheet')
...     worksheet = await sht.get_worksheet('Cool worksheet!')
```

id
Spreadsheet ID.

list_permissions()
This function is a *coroutine*.

Lists the spreadsheet's permissions.

remove_permissions(*value*, *role='any'*)
This function is a *coroutine*.

Example:

```
# Remove Otto's write permission for this spreadsheet
await sh.remove_permissions('otto@example.com', role='writer')

# Remove all Otto's permissions for this spreadsheet
await sh.remove_permissions('otto@example.com')
```

share(*value*, *perm_type*, *role*, *notify=True*, *email_message=None*)
This function is a *coroutine*.

Share the spreadsheet with other accounts. :param value: user or group e-mail address, domain name

or None for 'default' type.

Parameters

- **perm_type** – the account type. Allowed values are: `user`, `group`, `domain`, `anyone`.
- **role** – the primary role for this user. Allowed values are: `owner`, `writer`, `reader`.
- **notify** – Whether to send an email to the target user/domain.
- **email_message** – The email to be sent if notify=True

Example:

```
# Give Otto a write permission on this spreadsheet
await sh.share('otto@example.com', perm_type='user', role='writer')

# Transfer ownership to Otto
await sh.share('otto@example.com', perm_type='user', role='owner')
```

sheet1
Shortcut property for getting the first worksheet.

title
Spreadsheet title.

worksheet(*title*)
This function is a *coroutine*.

Returns a worksheet with specified *title*.

Parameters **title** – A title of a worksheet. If there're multiple worksheets with the same title, first one will be returned.

Returns an instance of `gsperad.models.Worksheet`.

Example. Getting worksheet named 'Annual bonuses'

```
>>> async def sheet():
...     sht = await client.open('Sample one')
...     worksheet = await sht.worksheet('Annual bonuses')
```

worksheets()
This function is a *coroutine*.

Returns a list of all `worksheets` in a spreadsheet.

class aiospread.models.**Worksheet**(*spreadsheet*, *properties*)
The class that represents a single sheet in a spreadsheet (aka "worksheet").

acell(*label*, *value_render_option='FORMATTED_VALUE'*)
This function is a *coroutine*.

Returns an instance of a `gspread.models.Cell`.

Parameters

- **label** – String with cell label in common format, e.g. 'B1'. Letter case is ignored.
- **value_render_option** – Determines how values should be rendered in the the output. See ValueRenderOption in the Sheets API.

Example:

```
>>> async def cell():
...     await worksheet.acell('A1')
<Cell R1C1 "I'm cell A1">
```

add_cols(*cols*)
This function is a *coroutine*.

Adds colums to worksheet.

Parameters **cols** – Columns number to add.

add_rows(*rows*)
This function is a *coroutine*.

Adds rows to worksheet.

Parameters **rows** – Rows number to add.

append_row(*values*, *value_input_option='RAW'*)
This function is a *coroutine*.

Adds a row to the worksheet and populates it with values. Widens the worksheet if there are more values than columns.

Parameters **values** – List of values for the new row.

cell(*row*, *col*, *value_render_option='FORMATTED_VALUE'*)
This function is a *coroutine*.

Returns an instance of a `gspread.models.Cell` positioned in *row* and *col* column.

Parameters

- **row** – Integer row number.
- **col** – Integer column number.
- **value_render_option** – Determines how values should be rendered in the the output. See ValueRenderOption in the Sheets API.

Example:

```
>>> async def cell():
...     await worksheet.cell(1, 1)
<Cell R1C1 "I'm cell A1">
```

clear()
This function is a *coroutine*.

Clears all cells in the worksheet.

col_count
Number of columns.

col_values(*col*, *value_render_option='FORMATTED_VALUE'*)
This function is a *coroutine*.

Returns a list of all values in column *col*.

Empty cells in this list will be rendered as `None`.

Parameters

- **col** – Integer column number.
- **value_render_option** – Determines how values should be rendered in the the output. See ValueRenderOption in the Sheets API.

delete_row(*index*)
"This function is a *coroutine*.

Deletes a row from the worksheet at the specified index.

Parameters **index** – Index of a row for deletion.

find(*query*)
This function is a *coroutine*.

Finds first cell matching query.

Parameters **query** – A text string or compiled regular expression.

findall(*query*)
This function is a *coroutine*.

Finds all cells matching query.

Parameters **query** – A text string or compiled regular expression.

get_all_records(*empty2zero=False*, *head=1*, *default_blank=''*)
This function is a *coroutine*.

Returns a list of dictionaries, all of them having the contents of the spreadsheet with the head row as keys and each of these dictionaries holding the contents of subsequent rows of cells as values.

Cell values are numericised (strings that can be read as ints or floats are converted).

Parameters

- **empty2zero** – determines whether empty cells are converted to zeros.

- **head** – determines wich row to use as keys, starting from 1 following the numeration of the spreadsheet.
- **default_blank** – determines whether empty cells are converted to something else except empty string or zero.

get_all_values()
This function is a *coroutine*.

Returns a list of lists containing all cells' values as strings.

id
Id of a worksheet.

insert_row(*values*, *index=1*, *value_input_option='RAW'*)
This function is a *coroutine*.

Adds a row to the worksheet at the specified index and populates it with values.

Widens the worksheet if there are more values than columns.

Parameters

- **values** – List of values for the new row.
- **value_input_option** – Determines how input data should be interpreted. See ValueInputOption in the Sheets API.

range(*name*)
This function is a *coroutine*.

Returns a list of *Cell* objects from a specified range.

Parameters **name** – A string with range value in A1 notation, e.g. 'A1:A5'.

Alternatively, you may specify numeric boundaries. All values index from 1 (one):

Parameters

- **first_row** – Integer row number
- **first_col** – Integer row number
- **last_row** – Integer row number
- **last_col** – Integer row number

Example::

```
>>> async def range():
...     # Using A1 notation
...     worksheet.range('A1:B7')
[<Cell R1C1 "42">, ...]
```

```
>>> async def range2():
...     # Same with numeric boundaries
...     worksheet.range(1, 1, 7, 2)
[<Cell R1C1 "42">, ...]
```

resize(*rows=None*, *cols=None*)
This function is a *coroutine*.

Resizes the worksheet.

Parameters

- **rows** – New rows number.
- **cols** – New columns number.

row_count
Number of rows.

row_values(*row*, *value_render_option='FORMATTED_VALUE'*)
This function is a *coroutine*.

Returns a list of all values in a *row*.

Empty cells in this list will be rendered as `None`.

Parameters

- **row** – Integer row number.
- **value_render_option** – Determines how values should be rendered in the the output. See ValueRenderOption in the Sheets API.

title
Title of a worksheet.

update_acell(*label*, *value*)
This function is a *coroutine*.

Sets the new value to a cell.

Parameters

- **label** – String with cell label in common format, e.g. 'B1'. Letter case is ignored.
- **value** – New value.

Example:

```
await worksheet.update_acell('A1', '42')
```

update_cell(*row*, *col*, *value*)
This function is a *coroutine*.

Sets the new value to a cell.

Parameters

- **row** – Row number.
- **col** – Column number.
- **value** – New value.

Example:

```
await worksheet.update_cell(1, 1, '42')
```

update_cells(*cell_list*, *value_input_option='RAW'*)
This function is a *coroutine*.

Updates cells in batch.

Parameters

- **cell_list** – List of a *Cell* objects to update.

- **value_input_option** – Determines how input data should be interpreted. See ValueInputOption in the Sheets API.

Example:

```
# Select a range
cell_list = await worksheet.range('A1:C7')

for cell in cell_list:
    cell.value = 'O_o'

# Update in batch
await worksheet.update_cells(cell_list)
```

update_title(*title*)

This function is a *coroutine*.

Renames the worksheet.

Parameters **title** – A new title.

updated

Deprecated since version 2.0.

This feature is not supported in Sheets API v4.

class aiospread.models.**Cell**(*row*, *col*, *value=''*)

An instance of this class represents a single cell in a `worksheet`.

col

Column number of the cell.

row

Row number of the cell.

value = None

Value of the cell.

CHAPTER 3

Utils

3.1 gspread.utils

This module contains utility functions.

aiospread.utils.**rowcol_to_a1**(*row*, *col*)

Translates a row and column cell address to A1 notation.

Parameters

- **row** – The row of the cell to be converted. Rows start at index 1.
- **col** – The column of the cell to be converted. Columns start at index 1.

Returns a string containing the cell's coordinates in A1 notation.

Example:

```
>>> rowcol_to_a1(1, 1)
A1
```

aiospread.utils.**a1_to_rowcol**(*label*)

Translates a cell's address in A1 notation to a tuple of integers.

Parameters **label** – String with cell label in A1 notation, e.g. 'B1'. Letter case is ignored.

Returns a tuple containing *row* and *column* numbers. Both indexed from 1 (one).

Example:

```
>>> a1_to_rowcol('A1')
(1, 1)
```

CHAPTER 4

Exceptions

exception aiospread.exceptions.**GSpreadException**
A base class for gspread's exceptions.

exception aiospread.exceptions.**APIError**(*response*)

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

aiospread.utils, 15

g

gspread, 1

Index

A

a1_to_rowcol() (*in module aiospread.utils*), 15
acell() (*aiospread.models.Worksheet method*), 9
add_cols() (*aiospread.models.Worksheet method*), 9
add_rows() (*aiospread.models.Worksheet method*), 9
add_worksheet() (*aiospread.models.Spreadsheet method*), 7
aiospread.utils (*module*), 15
APIError, 17
append_row() (*aiospread.models.Worksheet method*), 9

C

Cell (*class in aiospread.models*), 13
cell() (*aiospread.models.Worksheet method*), 9
clear() (*aiospread.models.Worksheet method*), 10
Client (*class in aiospread*), 3
col (*aiospread.models.Cell attribute*), 13
col_count (*aiospread.models.Worksheet attribute*), 10
col_values() (*aiospread.models.Worksheet method*), 10
create() (*aiospread.Client method*), 3

D

del_spreadsheet() (*aiospread.Client method*), 3
del_worksheet() (*aiospread.models.Spreadsheet method*), 7
delete_row() (*aiospread.models.Worksheet method*), 10

F

find() (*aiospread.models.Worksheet method*), 10
findall() (*aiospread.models.Worksheet method*), 10

G

get_all_records() (*aiospread.models.Worksheet method*), 10
get_all_values() (*aiospread.models.Worksheet method*), 11
get_worksheet() (*aiospread.models.Spreadsheet method*), 7
gspread (*module*), 1
GSpreadException, 17

I

id (*aiospread.models.Spreadsheet attribute*), 8
id (*aiospread.models.Worksheet attribute*), 11
import_csv() (*aiospread.Client method*), 4
insert_permission() (*aiospread.Client method*), 4
insert_row() (*aiospread.models.Worksheet method*), 11

L

list_permissions() (*aiospread.Client method*), 4
list_permissions() (*aiospread.models.Spreadsheet method*), 8
login() (*aiospread.Client method*), 4

O

open() (*aiospread.Client method*), 5
open_by_key() (*aiospread.Client method*), 5
open_by_url() (*aiospread.Client method*), 5
openall() (*aiospread.Client method*), 5

R

range() (*aiospread.models.Worksheet method*), 11
remove_permission() (*aiospread.Client method*), 5
remove_permissions() (*aiospread.models.Spreadsheet method*), 8
resize() (*aiospread.models.Worksheet method*), 11
row (*aiospread.models.Cell attribute*), 13
row_count (*aiospread.models.Worksheet attribute*), 12
row_values() (*aiospread.models.Worksheet method*), 12

rowcol_to_a1() (*in module aiospread.utils*), 15

S

share() (*aiospread.models.Spreadsheet method*), 8
sheet1 (*aiospread.models.Spreadsheet attribute*), 8
Spreadsheet (*class in aiospread.models*), 7

T

title (*aiospread.models.Spreadsheet attribute*), 8
title (*aiospread.models.Worksheet attribute*), 12

U

update_acell() (*aiospread.models.Worksheet method*), 12
update_cell() (*aiospread.models.Worksheet method*), 12
update_cells() (*aiospread.models.Worksheet method*), 12
update_title() (*aiospread.models.Worksheet method*), 13
updated (*aiospread.models.Worksheet attribute*), 13

V

value (*aiospread.models.Cell attribute*), 13

W

Worksheet (*class in aiospread.models*), 9
worksheet() (*aiospread.models.Spreadsheet method*), 8
worksheets() (*aiospread.models.Spreadsheet method*), 9