

---

# **gsread Documentation**

*Release 0.6.2*

**Anton Burnashev**

**Jul 13, 2017**



---

## Contents

---

|          |                               |           |
|----------|-------------------------------|-----------|
| <b>1</b> | <b>Main Interface</b>         | <b>3</b>  |
| <b>2</b> | <b>Models</b>                 | <b>7</b>  |
| <b>3</b> | <b>Utils</b>                  | <b>13</b> |
| 3.1      | gspread.utils . . . . .       | 13        |
| <b>4</b> | <b>Exceptions</b>             | <b>15</b> |
| <b>5</b> | <b>Internal Modules</b>       | <b>17</b> |
| 5.1      | gspread.httpsession . . . . . | 17        |
| 5.2      | gspread.urls . . . . .        | 17        |
| <b>6</b> | <b>Indices and tables</b>     | <b>19</b> |
|          | <b>Python Module Index</b>    | <b>21</b> |



gsread is a Python client library for the Google Sheets API.

- *Main Interface*
- *Models*
- *Utils*
  - *gsread.utils*
- *Exceptions*
- *Internal Modules*
  - *gsread.httpsession*
  - *gsread.urls*



---

## Main Interface

---

`gspread.authorize(credentials)`

Login to Google API using OAuth2 credentials. This is a shortcut function which instantiates *Client* and performs login right away. :returns: *Client* instance.

**class** `gspread.Client(auth, http_session=None)`

An instance of this class communicates with Google Data API.

### Parameters

- **auth** – An OAuth2 credential object. Credential objects are those created by the `oauth2client` library. <https://github.com/google/oauth2client>
- **http\_session** – (optional) A session object capable of making HTTP requests while persisting headers. Defaults to *HTTPSession*.

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

**create** (*title*)

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*)

Deletes a spreadsheet.

**Parameters** *file\_id* – a spreadsheet ID (aka file ID.)

**import\_csv** (*file\_id*, *data*)

Imports data into the first page of the spreadsheet.

**Parameters** *data* – A CSV string of data.

**insert\_permission** (*file\_id*, *value*, *perm\_type*, *role*, *notify=True*, *email\_message=None*)

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_6dprnRRUW11UFE',
    'otto@example.org',
    perm_type='user',
    role='writer'
)

# Make the spreadsheet publicly readable

gc.insert_permission(
    '0BmgG6nO_6dprnRRUW11UFE',
    None,
    perm_type='anyone',
    role='reader'
)
```

**list\_permissions** (*file\_id*)

Retrieve a list of permissions for a file.

**Parameters** *file\_id* – a spreadsheet ID (aka file ID.)

**login** ()

Authorize client.

**open** (*title*)

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

**open\_by\_key** (*key*)

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.

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

```
>>> c = gspread.authorize(credentials)
>>> c.open_by_key('0BmgG6nO_6dprdS1MN3d3MkdPa142WFRrdnRRUW11UFE')
```

**open\_by\_url** (*url*)

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

**openall** (*title=None*)

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.



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** `gsread.Spreadsheet` (*client, feed\_entry*)

A class for a spreadsheet object.

**add\_worksheet** (*title, rows, cols*)

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*)

Deletes a worksheet from a spreadsheet.

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

**get\_worksheet** (*index*)

Returns a worksheet with specified *index*.

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

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

Example. To get first worksheet of a spreadsheet:

```
>>> sht = client.open('My fancy spreadsheet')
>>> worksheet = sht.get_worksheet(0)
```

**id**

Spreadsheet ID.

**list\_permissions()**

Lists the spreadsheet's permissions.

**remove\_permissions** (*value*, *role='any'*)

Example:

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

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

**share** (*value*, *perm\_type*, *role*, *notify=True*, *email\_message=None*)

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
sh.share('otto@example.com', perm_type='user', role='writer')

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

**sheet1**

Shortcut property for getting the first worksheet.

**title**

Spreadsheet title.

**updated**

Updated time in RFC 3339 format

**worksheet** (*title*)

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 *Worksheet*.

Example. Getting worksheet named 'Annual bonuses'

```
>>> sht = client.open('Sample one')
>>> worksheet = sht.worksheet('Annual bonuses')
```

**worksheets** ()

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

**class** `gspread.Worksheet` (*spreadsheet, element*)

A class for worksheet object.

**acell** (*label*)

Returns an instance of a `Cell`.

**Parameters** `label` – String with cell label in common format, e.g. ‘B1’. Letter case is ignored.

Example:

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

**add\_cols** (*cols*)

Adds columns to worksheet.

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

**add\_rows** (*rows*)

Adds rows to worksheet.

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

**append\_row** (*values*)

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

Note that a new Google Sheet has 100 or 1000 rows by default. You may need to scroll down to find the new row.

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

**cell** (*row, col*)

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

**Parameters**

- `row` – Integer row number.
- `col` – Integer column number.

Example:

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

**clear** ()

Clears all cells in the worksheet.

**col\_count**

Number of columns

**col\_values** (*col*)

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

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

**delete\_row** (*index*)

“Deletes a row from the worksheet at the specified index

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

**export** (*format='csv'*)

Export the worksheet in specified format.

**Parameters** **format** – A format of the output.

**find** (*query*)

Finds first cell matching query.

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

**findall** (*query*)

Finds all cells matching query.

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

**get\_addr\_int** (*row, col*)

Translates cell's tuple of integers to a cell label.

Deprecated since version 0.5: Use `utils.rowcol_to_a1()` instead.

**get\_all\_records** (*empty2zero=False, head=1, default\_blank=''*)

**Returns a list of dictionaries, all of them having:**

- the contents of the spreadsheet's 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 which 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** ()

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

**get\_int\_addr** (*label*)

Translates cell's label address to a tuple of integers.

Deprecated since version 0.5: Use `utils.a1_to_rowcol()` instead.

**gid**

Gid of a worksheet.

**id**

Id of a worksheet.

**insert\_row** (*values, index=1*)

“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.

**range** (*\*args, \*\*kwargs*)

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:

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

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

**resize** (*rows=None, cols=None*)

Resizes the worksheet.

**Parameters**

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

**row\_count**

Number of rows

**row\_values** (*row*)

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

Empty cells in this list will be rendered as None.

**title**

Title of a worksheet.

**update\_acell** (*label, val*)

Sets the new value to a cell.

**Parameters**

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

Example:

```
worksheet.update_acell('A1', '42') # this could be 'a1' as well
```

**update\_cell** (*row, col, val*)

Sets the new value to a cell.

**Parameters**

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

Example:

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

**update\_cells** (*cell\_list*)

Updates cells in batch.

**Parameters** **cell\_list** – List of a *Cell* objects to update.

Example:

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

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

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

**update\_title** (*title*)

Renames the worksheet.

**Parameters** **title** – A new title.

**updated**

Updated time in RFC 3339 format

**class** `gsread.Cell` (*worksheet, element*)

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

**col**

Column number of the cell.

**input\_value = None**

Raw value of the cell (e.g. formula)

**row**

Row number of the cell.

**value = None**

Value of the cell.



## gsread.utils

This module contains utility functions.

`gsread.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
```

`gsread.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** `gsread.utils.NoValidUrlKeyFound`  
No valid key found in URL.



Following modules are for internal use only.

### **gsread.httpsession**

This module contains a class for working with http sessions.

**class** `gsread.httpsession.HTTPSession` (*headers=None*)  
Handles HTTP activity while keeping headers persisting across requests.  
**Parameters** **headers** – A dict with initial headers.

### **gsread.urls**

This module is Google API url patterns storage.

`gsread.urls.construct_url` (*feedtype=None, obj=None, visibility='private', projection='full', spreadsheet\_id=None, worksheet\_id=None, cell\_id=None, worksheet\_version=None*)  
Constructs URL to be used for API request.



## CHAPTER 6

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`





**g**

gsread, 1  
gsread.httpsession, 17  
gsread.urls, 17  
gsread.utils, 13



**A**

`a1_to_rowcol()` (in module `gspread.utils`), 13  
`acell()` (`gspread.Worksheet` method), 9  
`add_cols()` (`gspread.Worksheet` method), 9  
`add_rows()` (`gspread.Worksheet` method), 9  
`add_worksheet()` (`gspread.Spreadsheet` method), 7  
`append_row()` (`gspread.Worksheet` method), 9  
`authorize()` (in module `gspread`), 3

**C**

`Cell` (class in `gspread`), 12  
`cell()` (`gspread.Worksheet` method), 9  
`clear()` (`gspread.Worksheet` method), 9  
`Client` (class in `gspread`), 3  
`col` (`gspread.Cell` attribute), 12  
`col_count` (`gspread.Worksheet` attribute), 9  
`col_values()` (`gspread.Worksheet` method), 9  
`construct_url()` (in module `gspread.urls`), 17  
`create()` (`gspread.Client` method), 3

**D**

`del_spreadsheet()` (`gspread.Client` method), 4  
`del_worksheet()` (`gspread.Spreadsheet` method), 7  
`delete_row()` (`gspread.Worksheet` method), 9

**E**

`export()` (`gspread.Worksheet` method), 9

**F**

`find()` (`gspread.Worksheet` method), 10  
`findall()` (`gspread.Worksheet` method), 10

**G**

`get_addr_int()` (`gspread.Worksheet` method), 10  
`get_all_records()` (`gspread.Worksheet` method), 10  
`get_all_values()` (`gspread.Worksheet` method), 10  
`get_int_addr()` (`gspread.Worksheet` method), 10  
`get_worksheet()` (`gspread.Spreadsheet` method), 7  
`gid` (`gspread.Worksheet` attribute), 10

`gspread` (module), 1  
`gspread.httpsession` (module), 17  
`gspread.urls` (module), 17  
`gspread.utils` (module), 13

**H**

`HTTPSession` (class in `gspread.httpsession`), 17

**I**

`id` (`gspread.Spreadsheet` attribute), 7  
`id` (`gspread.Worksheet` attribute), 10  
`import_csv()` (`gspread.Client` method), 4  
`input_value` (`gspread.Cell` attribute), 12  
`insert_permission()` (`gspread.Client` method), 4  
`insert_row()` (`gspread.Worksheet` method), 10

**L**

`list_permissions()` (`gspread.Client` method), 4  
`list_permissions()` (`gspread.Spreadsheet` method), 8  
`login()` (`gspread.Client` method), 4

**N**

`NoValidUrlKeyFound`, 15

**O**

`open()` (`gspread.Client` method), 4  
`open_by_key()` (`gspread.Client` method), 5  
`open_by_url()` (`gspread.Client` method), 5  
`openall()` (`gspread.Client` method), 5

**R**

`range()` (`gspread.Worksheet` method), 10  
`remove_permission()` (`gspread.Client` method), 5  
`remove_permissions()` (`gspread.Spreadsheet` method), 8  
`resize()` (`gspread.Worksheet` method), 11  
`row` (`gspread.Cell` attribute), 12  
`row_count` (`gspread.Worksheet` attribute), 11  
`row_values()` (`gspread.Worksheet` method), 11  
`rowcol_to_a1()` (in module `gspread.utils`), 13

## S

share() (gsread.Spreadsheet method), 8  
sheet1 (gsread.Spreadsheet attribute), 8  
Spreadsheet (class in gsread), 7

## T

title (gsread.Spreadsheet attribute), 8  
title (gsread.Worksheet attribute), 11

## U

update\_acell() (gsread.Worksheet method), 11  
update\_cell() (gsread.Worksheet method), 11  
update\_cells() (gsread.Worksheet method), 12  
update\_title() (gsread.Worksheet method), 12  
updated (gsread.Spreadsheet attribute), 8  
updated (gsread.Worksheet attribute), 12

## V

value (gsread.Cell attribute), 12

## W

Worksheet (class in gsread), 8  
worksheet() (gsread.Spreadsheet method), 8  
worksheets() (gsread.Spreadsheet method), 8