
gsread-pandas Documentation

Release 3.2.2

Diego Fernandez

Oct 07, 2022

Contents

1	Overview	3
2	Installation / Usage	5
2.1	Client Credentials	5
3	Example	7
4	Troubleshooting	9
4.1	EOFError in Rodeo	9
4.2	This action would increase the number of cells in the workbook above the limit of 10000000 cells.	9
5	Configuration	11
5.1	App Credentials	11
5.2	User Credentials	11
5.3	Authentication	12
5.4	Alternate Workflows	12
6	Using Gspread-Pandas	13
6.1	Client	13
6.2	Spread	14
7	Modules	15
7.1	gspread_pandas	15
7.1.1	Submodules	15
7.1.1.1	gspread_pandas.spread module	15
7.1.1.2	gspread_pandas.client module	21
7.1.1.3	gspread_pandas.conf module	23
8	Contributing	25
8.1	Testing	25
8.2	Versions	25
8.3	CI	26
9	Change Log	27
9.1	[Unreleased]	27
9.2	[3.2.2] - 2022-06-29	27
9.3	Changed	27
9.4	[3.2.1] - 2022-06-29	27

9.5	Fixed	27
9.6	Removed	27
9.7	[3.2.0] - 2022-03-19	28
9.8	[3.1.0] - 2022-03-19	28
9.9	Fixed	28
9.10	[3.0.4] - 2022-01-18	28
9.11	Fixed	28
9.12	[3.0.3] - 2022-01-04	28
9.13	Fixed	28
9.14	Added	28
9.15	[3.0.2] - 2021-12-29	28
9.16	Removed	28
9.17	[3.0.1] - 2021-12-28	28
9.18	Changed	28
9.19	[3.0.0] - 2021-12-28	29
9.20	Changed	29
9.21	Removed	29
9.22	Added	29
9.23	[2.3.1] - 2021-11-29	29
9.24	Fixed	29
9.25	[2.3.0] - 2021-03-21	29
9.26	Changed	29
9.27	Fixed	29
9.28	Added	29
9.29	[2.2.4] - 2021-01-24	30
9.30	Fixed	30
9.31	[2.2.3] - 2020-03-26	30
9.32	Fixed	30
9.33	[2.2.2] - 2020-03-21	30
9.34	Changed	30
9.35	[2.2.1] - 2020-01-05	30
9.36	Added	30
9.37	Fixed	30
9.38	Changed	30
9.39	[2.2.0] - 2019-11-17	31
9.40	Added	31
9.41	Fixed	31
9.42	Deprecated	31
9.43	Changed	31
9.44	[2.1.3] - 2019-08-25	31
9.45	Added	31
9.46	Changed	31
9.47	[2.1.2] - 2019-07-10	32
9.48	Added	32
9.49	Fixed	32
9.50	Changed	32
9.51	[2.1.1] - 2019-07-10	32
9.52	Fixed	32
9.53	[2.1.0] - 2019-07-10	32
9.54	Added	32
9.55	Changed	32
9.56	Fixed	32
9.57	[2.0.0] - 2019-06-12	33
9.58	Added	33

9.59	Removed	33
9.60	Changed	33
9.61	Fixed	33
9.62	[1.3.1] - 2019-05-17	33
9.62.1	Fixed	33
9.63	[1.3.0] - 2019-04-30	34
9.63.1	Added	34
9.63.2	Fixed	34
9.63.3	Deprecated	34
9.63.4	Changed	34
9.64	[1.2.2] - 2019-04-15	34
9.64.1	Fixed	34
9.65	[1.2.1] - 2018-08-30	35
9.65.1	Fixed	35
9.66	[1.2.0] - 2018-08-30	35
9.66.1	Added	35
9.66.2	Fixed	35
9.66.3	Changed	35
9.67	[1.1.3] - 2018-07-07	35
9.67.1	Added	35
9.67.2	Fixed	35
9.68	[1.1.2] - 2018-06-23	35
9.68.1	Fixed	35
9.68.2	Changed	36
9.69	[1.1.1] - 2018-06-13	36
9.69.1	Changed	36
9.70	[1.1.0] - 2018-06-02	36
9.70.1	Fixed	36
9.70.2	Changed	36
9.70.3	Added	36
9.71	[1.0.5] - 2018-04-14	36
9.71.1	Fixed	36
9.72	[1.0.4] - 2018-04-08	36
9.72.1	Fixed	36
9.73	[1.0.3] - 2018-04-02	37
9.73.1	Added	37
9.74	[1.0.2] - 2018-04-02	37
9.74.1	Changed	37
9.75	[1.0.1] - 2018-03-26	37
9.75.1	Changed	37
9.76	[1.0.0] - 2018-03-26	37
9.76.1	Added	37
9.76.2	Changed	37
9.77	[0.16.1] - 2018-03-24	38
9.77.1	Fixed	38
9.78	[0.16.0] - 2018-03-24	38
9.78.1	Added	38
9.78.2	Fixed	38
9.78.3	Changed	38
9.79	[0.15.6] - 2018-03-12	38
9.79.1	Fixed	38
9.80	[0.15.5] - 2018-03-12	38
9.80.1	Fixed	38
9.81	[0.15.4] - 2018-02-13	39

9.81.1	Fixed	39
9.81.2	Added	39
9.82	[0.15.3] - 2017-11-21	39
9.82.1	Changed	39
9.83	[0.15.2] - 2017-11-18	39
9.83.1	Fixed	39
9.83.2	Changed	39
9.84	[0.15.1] - 2017-10-05	39
9.84.1	Fixed	39
9.85	[0.15.0] - 2017-09-11	39
9.85.1	Changed	39
9.85.2	Fixed	40
9.86	[0.14.3] - 2017-06-22	40
9.86.1	Changed	40
9.87	[0.14.2] - 2017-06-18	40
9.87.1	Added	40
9.87.2	Fixed	40
9.88	[0.14.1] - 2017-06-05	40
9.89	Changed	40
9.90	[0.14.0] - 2017-05-25	40
9.90.1	Added	40
9.90.2	Changed	41
9.90.3	Fixed	41
9.91	[0.13.0] - 2017-04-28	41
9.91.1	Added	41
9.92	[0.12.1] - 2017-04-25	41
9.92.1	Changed	41
9.92.2	Fixed	41
9.93	[0.12.0] - 2017-03-31	41
9.93.1	Added	41
9.93.2	Fixed	41
9.94	[0.11.2] - 2017-03-22	42
9.94.1	Changed	42
9.95	[0.11.1] - 2017-03-22	42
9.95.1	Added	42
9.95.2	Changed	42
9.95.3	Fixed	42
9.96	[0.11.0] - 2017-02-14	42
9.96.1	Changed	42
9.96.2	Fixed	42
9.97	[0.10.1] - 2017-01-26	42
9.97.1	Added	42
9.97.2	Changed	42
9.98	[0.10.0] - 2017-01-18	43
9.98.1	Added	43
9.98.2	Changed	43
9.98.3	Fixed	43
9.98.4	Deprecated	43
9.99	[0.9] - 2016-12-07	43
9.99.1	Added	43
9.99.2	Changed	44
9.99.3	Fixed	44
9.100	[0.8] - 2016-11-11	44
9.100.1	Added	44

9.100.2 Fixed	44
9.101 [0.7] - 2016-11-10	44
9.101.1 Changed	44
9.102 [0.6] - 2016-10-27	44
9.102.1 Changed	44
9.103 [0.5] - 2016-10-19	44
9.103.1 Changed	44
9.104 [0.4] - 2016-10-19	45
9.104.1 Added	45
9.105 Changed	45
9.106 [0.3] - 2016-10-19	45
9.106.1 Changed	45
9.107 [0.2] - 2016-10-12	45
9.107.1 Added	45
9.108 [0.1] - 2016-10-11	45
9.108.1 Added	45
10 Indices and tables	47
Python Module Index	49
Index	51

author: Diego Fernandez

Links:

- [Documentation](#)
- [Source code](#)
- [Short video tutorial](#)

Attention: Upgrading from < 2.0

If you are upgrading, the `user` is now an optional param that uses `default` as the default. If you're a single user, you might want to re-name your credentials to `default` so you can stop specifying it:

```
mv ~/.config/gspread_pandas/creds{<old_name>,default}
```


A package to easily open an instance of a Google spreadsheet and interact with worksheets through Pandas DataFrames. It enables you to easily pull data from Google spreadsheets into DataFrames as well as push data into spreadsheets from DataFrames. It leverages `gsread` in the backend for most of the heavy lifting, but it has a lot of added functionality to handle things specific to working with DataFrames as well as some extra nice to have features.

The target audience are Data Analysts and Data Scientists, but it can also be used by Data Engineers or anyone trying to automate workflows with Google Sheets and Pandas.

Some key goals/features:

- Be easy to use interactively, with good docstrings and auto-completion
- Nicely handle headers and indexes (including multi-level headers and merged cells)
- Run on Jupyter, headless server, and/or scripts
- Allow storing different user credentials or using Service Accounts
- Automatically handle token refreshes
- Enable handling of frozen rows and columns
- Enable filling in all merged cells when pulling data
- Nicely handle large data sets and auto-retries
- Enable creation of filters
- Handle retries when exceeding 100 second user quota
- When pushing DataFrames with MultiIndex columns, allow merging or flattening headers
- Ability to handle Spreadsheet permissions
- Ability to specify `ValueInputOption` and `ValueRenderOption` for specific columns

To install use pip:

```
$ pip install gspread-pandas
```

Or clone the repo:

```
$ git clone https://github.com/aiguofer/gspread-pandas.git
$ python setup.py install
```

Before using, you will need to download Google client credentials for your app.

2.1 Client Credentials

To allow a script to use Google Drive API we need to authenticate our self towards Google. To do so, we need to create a project, describing the tool and generate credentials. Please use your web browser and go to [Google console](#) and :

- Choose **Create Project** in popup menu on the top.
- A dialog box appears, so give your project a name and click on **Create** button.
- On the left-side menu click on **API Manager**.
- A table of available APIs is shown. Switch **Drive API** and click on **Enable API** button. Do the same for **Sheets API**. Other APIs might be switched off, for our purpose.
- On the left-side menu click on **Credentials**.
- In section **OAuth consent screen** select your email address and give your product a name. Then click on **Save** button.
- In section **Credentials** click on **Add credentials** and switch **OAuth client ID** (if you want to use your own account or enable the use of multiple accounts) or **Service account key** (if you prefer to have a service account interacting with spreadsheets).

- If you select **OAuth client ID**:
 - Select **Application type** item as **Desktop app** and give it a name.
 - Click on **Create** button.
 - Click on **Download JSON** icon on the right side of created **OAuth client IDs** and store the downloaded file on your file system.
- If you select **Service account key**
 - Click on **Service account** dropdown and select **New service account**
 - Give it a **Service account name** and ignore the **Role** dropdown (unless you know you need this for something else, it's not necessary for working with spreadsheets)
 - Note the **Service account ID** as you might need to give that user permission to interact with your spreadsheets
 - Leave **Key type** as **JSON**
 - Click **Create** and store the downloaded file on your file system.
- Please be aware, the file contains your private credentials, so take care of the file in the same way you care of your private SSH key; Move the downloaded JSON to `~/.config/gspread_pandas/google_secret.json` (or you can configure the directory and file name by directly calling `gspread_pandas.conf.get_config`)

Thanks to similar project [df2gspread](#) for this great description of how to get the client credentials.

You can read more about it in the [configuration docs](#) including how to change the default behavior.

CHAPTER 3

Example

```
import pandas as pd
from gspread_pandas import Spread, Client

file_name = "http://stats.idre.ucla.edu/stat/data/binary.csv"
df = pd.read_csv(file_name)

# 'Example Spreadsheet' needs to already exist and your user must have access to it
spread = Spread('Example Spreadsheet')
# This will ask to authenticate if you haven't done so before

# Display available worksheets
spread.sheets

# Save DataFrame to worksheet 'New Test Sheet', create it first if it doesn't exist
spread.df_to_sheet(df, index=False, sheet='New Test Sheet', start='A2', replace=True)
spread.update_cells('A1', 'B1', ['Created by:', spread.email])
print(spread)
# <gspread_pandas.client.Spread - User: '<example_user>@gmail.com', Spread: 'Example_
->Spreadsheet', Sheet: 'New Test Sheet'>

# You can now first instantiate a Client separately and query folders and
# instantiate other Spread objects by passing in the Client
client = Client()
# Assuming you have a dir called 'example dir' with sheets in it
available_sheets = client.find_spreadsheet_files_in_folders('example dir')
spreads = []
for sheet in available_sheets.get('example dir', []):
    spreads.append(Spread(sheet['id'], client=client))
```


4.1 EOFError in Rodeo

If you're trying to use `gsread_pandas` from within Rodeo you might get an `EOFError: EOF` when reading a line error when trying to pass in the verification code. The workaround for this is to first verify your account in a regular shell. Since you're just doing this to get your OAuth token, the spreadsheet doesn't need to be valid. Just run this in shell:

```
python -c "from gsread_pandas import Spread; Spread('<user_key>', '')"
```

Then follow the instructions to create and store the OAuth creds.

4.2 This action would increase the number of cells in the workbook above the limit of 10000000 cells.

IMO, Google sheets is not the right tool for large datasets. However, there's probably good reasons you might have to use it in such cases. When uploading a large DataFrame, you might run into this error.

By default, `Spread.df_to_sheet` will add rows and/or columns needed to accommodate the DataFrame. Since a new sheet contains a fairly large number of columns, if you're uploading a DF with lots of rows you might exceed the max number of cells in a worksheet even if your data does not. In order to fix this you have 2 options:

1. The easiest is to pass `replace=True`, which will first resize the worksheet and clear out all values.
2. Another option is to first resize to 1x1 using `Spread.sheet.resize(1, 1)` and then do `df_to_sheet`

There's a strange caveat with resizing, so going to 1x1 first is recommended (`replace=True` already does this). To read more see [this issue](#)

By default, the configuration will be in `$HOME/.config/gspread_pandas` on Nix systems and `%APPDATA%\gspread_pandas` on Windows. Under the default behavior, you must have your Google client credentials stored in `google_secret.json` in that directory. If you're not using a Service Account, the user credentials will be stored in a subdirectory called `creds`.

5.1 App Credentials

There's 2 main types of app credentials: OAuth client and Service Account. In order to act as your own Google user, you will need the OAuth client app credentials. With this type of credentials, each user will need to grant permissions to your app. When they grant permissions, their credentials will be stored as described below.

As a Service Account, the used credentials will be for the service account itself. This means that you'll be using the service account's e-mail and Google drive. Additionally, it will only be able to work with Spreadsheets that it has permissions for. Although Service Accounts can be useful for batch processes, you might generally prefer to work as your own user.

5.2 User Credentials

Once you have your client credentials, you can have multiple user credentials stored in the same machine. This can be useful when you have a shared server (for example with a Jupyter notebook server) with multiple people that may want to use the library. The `user` parameter to `Spread` must be the key identifying a user's credentials, by default it will store the creds using `default` as the key. The first `get_creds` is called for a specific key, you will have to authenticate through a text based OAuth prompt; this makes it possible to run on a headless server through ssh or through a Jupyter notebook. After this, the credentials for that user will be stored in the `creds` subdirectory and the tokens will be refreshed automatically any time the tool is used.

Users will only be able to interact with Spreadsheets that they have access to.

5.3 Authentication

In the backend, the library is leveraging Google's `google-auth` to handle authentication. It conveniently stores everything as described above so that you don't have to worry about boiler plate code to handle auth.

When a `Client` is instantiated, an `AuthorizedSession` is created using the credentials and this is what's used to make requests to the API. This takes care of handling token refreshes and retries for you.

5.4 Alternate Workflows

There's a variety of ways to change the default behavior of how/where configuration is stored.

The easiest way to change the default location is to set the `GSPREAD_PANDAS_CONFIG_DIR` env variable to the directory where you want to store everything. If you use this, the client creds will still need to be named `google_secret.json` and user creds will still be stored in the `creds` subdirectory.

If you have different client credentials, you could load them passing in `conf_dir` and/or `file_name` to `gsread_pandas.conf.get_config`. Alternatively, you could pull these from elsewhere, like a database. Once you have the config, you could then pass that to a `Client` or `Spread` instance, or you could get credentials by passing it to `gsread_pandas.conf.get_creds`.

When using a Service Account, the `user` param will be ignored in `Client`, `Spread` and `get_creds`. Otherwise, this param will be used to store the OAuth2 credentials for each user in the `creds` subdirectory. If you generate your credentials elsewhere, you can pass them in to a `Client` or `Spread`. You can also run through the flow to get OAuth2 and avoid saving them by calling `get_creds` directly. You can also override the `creds_dir` if you call this function.

Using Gspread-Pandas

There are two main objects you will interact with in `gspread-pandas`: the `Client` and the `Spread` objects. The goal of these objects is to make it easy to work with a variety of concepts in Google Sheets and Pandas DataFrames. A lot of care has gone into documenting functions in code to make code introspection tools useful (displaying documentation, code completion, etc).

The target audience are Data Analysts and Data Scientists, but this can also be used by Data Engineers or anyone trying to automate workflows with Google Sheets and Pandas.

6.1 Client

The `Client` extends the `Client` object from `gspread` to add some functionality. I try to contribute back to the upstream project, but some things don't make it in, and others don't belong.

The main things that are added by the `Client` are:

- Handling credentials and authentication to reduce boilerplate code.
- Store file paths within drive for more working with files in a more intuitive manner (requires passing `load_dirs=True` or calling `Client.refresh_directories()` if you've already instantiated a `Client`)
- A variety of functions to query for and work with Spreadsheets in your Google Drive, mainly:
 - `list_spreadsheet_files`
 - `list_spreadsheet_files_in_folder`
 - `find_folders`
 - `find_spreadsheet_files_in_folders`
 - `create_folder`
 - `move_file`
- Monkey patch the request to automatically retry when there is a 100 second quota exhausted error.

You can read more *in the docs for the Client object*.

6.2 Spread

The `Spread` object represents an open Google Spreadsheet. A `Spread` object has multiple `Worksheets`, and only one can be open at any one time. Any function you call will act on the currently open `Worksheet`, unless you pass `sheet=<worksheet_name_or_index>` when you call the function, in which case it will first open that `Worksheet` and then perform the action. A `Spread` object internally also holds an instance of a `Client` to do the majority of the work.

The `Spread` object does a lot of stuff to make it easier for working with Google Spreadsheets. For example, it can handle merged cells, frozen rows/columns, data filters, multi-level column headers, permissions, and more. Some things can be called individually, others can be passed in as function parameters. It can also work with tuples (for example `(1, 1)`) or A1 notation for specifying cells.

Some of the most important properties of a `Spread` object are:

- `spread`: The currently open Spreadsheet (this is a `gsread` object)
- `sheet`: The currently open `Worksheet` (this is a `gsread` object)
- `client`: The `Client` object. This will be automatically created if one is not passed in, but you can also share the same `Client` instance among multiple `Spread` objects if you pass it in.
- `sheets`: The list of all available `Worksheets`
- `_sheet_metadata`: We store metadata about the sheet, which includes stuff like merged cells, frozen columns, and frozen rows. This is a private property, but you can refresh this with `refresh_spread_metadata()`

Some of the most useful functions are:

- `sheet_to_df`: Create a Pandas `DataFrame` from a `Worksheet`
- `clear_sheet`: Clear out all values and resize a `Worksheet`
- `delete_sheet`: Delete a `Worksheet`
- `df_to_sheet`: Create a `Worksheet` from a Pandas `DataFrame`
- `freeze`: Freeze a given number of rows and/or columns
- `add_filter`: Add a filter to the `Worksheet` for the given range of data
- `merge_cells`: Merge cells in a `Worksheet`
- `unmerge_cells`: Unmerge cells within a range
- `add_permission`: Add a permission to a Spreadsheet
- `add_permissions`: Add multiple permissions to a Spreadsheet
- `list_permissions`: Show all current permissions on the Spreadsheet
- `move`: Move the Spreadsheet to a different location

You can read more *in the docs for the Spread object*.

7.1 gspread_pandas

7.1.1 Submodules

7.1.1.1 gspread_pandas.spread module

```
class gspread_pandas.spread.Spread(spread, sheet=0, config=None, create_spread=False,
                                     create_sheet=False, scope=['openid',
                                     'https://www.googleapis.com/auth/drive',
                                     'https://www.googleapis.com/auth/userinfo.email',
                                     'https://www.googleapis.com/auth/spreadsheets'],
                                     user='default', creds=None, client=None, permis-
                                     sions=None)
```

Bases: object

Simple wrapper for gspread to interact with Pandas. It holds an instance of an ‘open’ spreadsheet, an ‘open’ worksheet, and a list of available worksheets.

Each user will be associated with specific OAuth credentials. The authenticated user will need the appropriate permissions to the Spreadsheet in order to interact with it.

Parameters

- **spread** (*str*) – name, url, or id of the spreadsheet; must have read access by the authenticated user, see *open_spread*
- **sheet** (*str, int*) – optional, name or index of Worksheet, see *open_sheet* (default 0)
- **config** (*dict*) – optional, if you want to provide an alternate configuration, see *get_config* (default None)
- **create_sheet** (*bool*) – whether to create the worksheet if it doesn’t exist, it will use the spread value as the sheet title (default False)

- **create_spread** (*bool*) – whether to create the spreadsheet if it doesn't exist, it will use the `spread` value as the sheet title (default `False`)
- **scope** (*list*) – optional, if you'd like to provide your own scope (default `default_scope`)
- **user** (*str*) – string indicating the key to a user's credentials, which will be stored in a file (by default they will be stored in `~/.config/gspread_pandas/creds/<user>` but can be modified with `creds_dir` property in config). If using a Service Account, this will be ignored. (default `"default"`)
- **creds** (*google.auth.credentials.Credentials*) – optional, pass credentials if you have those already (default `None`)
- **client** (*Client*) – optional, if you've already instantiated a Client, you can just pass that and it'll be used instead (default `None`)
- **permissions** (*list*) – a list of strings. See [add_permissions](#) for the expected format

add_filter (*start=None, end=None, sheet=None*)

Add filters to data in the open worksheet.

Parameters

- **start** (*tuple, str*) – Tuple indicating (row, col) or string like 'A1' (default 'A1')
- **end** (*tuple, str*) – Tuple indicating (row, col) or string like 'A1' (default last cell in sheet)
- **sheet** (*str, int, Worksheet*) – optional, if you want to open or create a different sheet before adding the filter, see [open_sheet](#) (default `None`)

Returns

Return type `None`

add_permission (*permission*)

Add a permission to the current spreadsheet.

The format should be: `<id>|(<group>)|(<role>)|(<notify>)|(<require_link>)`
where:

`<id>` - email address of group or individual, domain, or 'anyone' `<group>` - optional, if the id is a group e-mail, this needs to be 'group' or

'grp'

`<role>` - optional, one of 'owner', 'writer', or 'reader'. If omitted, 'reader' will be used

`<notify>` - optional, if you don't want to notify the user, pass 'no' or 'false' `<require_link>` - optional, if you want to require the user to have the link,

pass 'link'

For example, to allow anyone with a link in the group `admins@example.com` to write when they have a link, but without sending a notification to the group: `admins@example.com|grp|owner|false|link`

Or if you want to give `user@example.com` reader permissions without a notification: `user@example.com|no`

Or to give anyone read access: `anyone`

Parameters **permissions** (*string*) – A string meeting the above mentioned format.

Returns**Return type** None**add_permissions** (*permissions*)

Add permissions to the current spreadsheet. See.

add_permission for format.**Parameters** **permissions** (*list*) – A list of strings meeting the above mentioned format.**Returns****Return type** None**clear_sheet** (*rows=1, cols=1, sheet=None*)

Reset open worksheet to a blank sheet with given dimensions.

Parameters

- **rows** (*int*) – number of rows (default 1)
- **cols** (*int*) – number of columns (default 1)
- **sheet** (*str, int, Worksheet*) – optional; name, index, or Worksheet, see *open_sheet* (default None)

Returns**Return type** None**create_sheet** (*name, rows=1, cols=1*)

Create a new worksheet with the given number of rows and cols.

Automatically opens that sheet after it's created.

Parameters

- **name** (*str*) – name of new Worksheet
- **rows** (*int*) – number of rows (default 1)
- **cols** (*int*) – number of columns (default 1)

Returns**Return type** None**delete_sheet** (*sheet*)Delete a worksheet by title. Returns whether the sheet was deleted or not. If current sheet is deleted, the *sheet* property will be set to None.**Parameters** **sheet** (*str, Worksheet*) – name or Worksheet**Returns** True if deleted successfully, else False**Return type** bool**df_to_sheet** (*df, index=True, headers=True, start=(1, 1), replace=False, sheet=None, raw_columns=None, freeze_index=False, freeze_headers=False, fill_value="", add_filter=False, merge_headers=False, flatten_headers_sep=None*)

Save a DataFrame into a worksheet.

Parameters

- **df** (*DataFrame*) – the DataFrame to save
- **index** (*bool*) – whether to include the index in worksheet (default True)

- **headers** (*bool*) – whether to include the headers in the worksheet (default True)
- **start** (*tuple, str*) – tuple indicating (row, col) or string like ‘A1’ for top left cell (default (1,1))
- **replace** (*bool*) – whether to remove everything in the sheet first (default False)
- **sheet** (*str, int, Worksheet*) – optional, if you want to open or create a different sheet before saving, see [open_sheet](#) (default None)
- **raw_columns** (*list, str*) – optional, list of columns from your dataframe that you want interpreted as RAW input in google sheets. This can be column names or column numbers.
- **freeze_index** (*bool*) – whether to freeze the index columns (default False)
- **freeze_headers** (*bool*) – whether to freeze the header rows (default False)
- **fill_value** (*str*) – value to fill nulls with (default ‘’)
- **add_filter** (*bool*) – whether to add a filter to the uploaded sheet (default False)
- **merge_headers** (*bool*) – whether to merge cells in the header that have the same value (default False)
- **flatten_headers_sep** (*str*) – if you want to flatten your multi-headers to a single row, you can pass the string that you’d like to use to concatenate the levels, for example, ‘: ‘ (default None)

Returns

Return type None

find_sheet (*sheet*)

Find a given worksheet by title or by object comparison.

Parameters **sheet** (*str, Worksheet*) – name of Worksheet or Worksheet object

Returns the Worksheet by the given name or None if not found

Return type Worksheet

freeze (*rows=None, cols=None, sheet=None*)

Freeze rows and/or columns for the open worksheet.

Parameters

- **rows** (*int*) – number of rows to freeze, use 0 to ‘unfreeze’ (default None)
- **cols** (*int*) – number of columns to freeze, use 0 to ‘unfreeze’ (default None)
- **sheet** (*str, int, Worksheet*) – optional, if you want to open or create a different sheet before freezing, see [open_sheet](#) (default None)

Returns

Return type None

get_sheet_dims (*sheet=None*)

Get the dimensions of the currently open Worksheet.

Parameters **sheet** (*str, int, Worksheet*) – optional, if you want to open a different sheet first, see [open_sheet](#) (default None)

Returns a tuple containing (num_rows,num_cols)

Return type tuple

list_permissions ()

List all permissions for this Spreadsheet.

Returns a list of dicts indicating the permissions on this spreadsheet

Return type list

merge_cells (*start*, *end*, *merge_type*='MERGE_ALL', *sheet*=None)

Merge cells between the start and end cells. Use *merge_type* if you want to change the behavior of the merge.

Parameters

- **start** (*tuple*, *str*) – Tuple indicating (row, col) or string like 'A1'
- **end** (*tuple*, *str*) – Tuple indicating (row, col) or string like 'A1'
- **merge_type** (*str*) – One of MERGE_ALL, MERGE_ROWS, or MERGE_COLUMNS (default "MERGE_ALL")
- **sheet** (*str*, *int*, *Worksheet*) – optional, if you want to open or create a different sheet before adding the filter, see *open_sheet* (default None)

Returns

Return type None

move (*path*='/', *create*=True)

Move the current spreadsheet to the specified path in your Google drive. If the file is not currently in you drive, it will be added.

Parameters

- **path** (*str*) – folder path (Default value = "/")
- **create** (*bool*) – if true, create folders as needed (Default value = True)

open (*spread*, *sheet*=None, *create_sheet*=False, *create_spread*=False)

Open a spreadsheet, and optionally a worksheet. See.

open_spread and *open_sheet*.

Parameters

- **spread** (*str*) – name, url, or id of Spreadsheet
- **sheet** (*str*, *int*) – name or index of Worksheet (default None)
- **create_sheet** (*bool*) – whether to create the worksheet if it doesn't exist, it will use the *spread* value as the sheet title (default False)
- **create_spread** (*bool*) – whether to create the spreadsheet if it doesn't exist, it will use the *spread* value as the sheet title (default False)

Returns

Return type None

open_sheet (*sheet*, *create*=False)

Open a worksheet. Optionally, if the sheet doesn't exist then create it first (only when *sheet* is a str).

Parameters

- **sheet** (*str*, *int*, *Worksheet*) – name, index, or Worksheet object
- **create** (*bool*) – whether to create the sheet if it doesn't exist, see *create_sheet* (default False)

Returns

Return type None

open_spread (*spread*, *create=False*)

Open a spreadsheet. Authorized user must already have read access.

Parameters

- **spread** (*str*) – name, url, or id of Spreadsheet
- **create** (*bool*) – whether to create the spreadsheet if it doesn't exist, it will use the `spread` value as the sheet title (default False)

Returns

Return type None

refresh_spread_metadata ()

Refresh spreadsheet metadata.

sheet_to_df (*index=1*, *header_rows=1*, *start_row=1*, *unformatted_columns=None*, *formula_columns=None*, *sheet=None*)

Pull a worksheet into a DataFrame.

Parameters

- **index** (*int*) – col number of index column, 0 or None for no index (default 1)
- **header_rows** (*int*) – number of rows that represent headers (default 1)
- **start_row** (*int*) – row number for first row of headers or data (default 1)
- **unformatted_columns** (*list*) – column numbers or names for columns you'd like to pull in as unformatted values (default [])
- **formula_columns** (*list*) – column numbers or names for columns you'd like to pull in as actual formulas (default [])
- **sheet** (*str*, *int*) – optional, if you want to open a different sheet first, see [open_sheet](#) (default None)

Returns DataFrame with the data from the Worksheet

Return type DataFrame

unmerge_cells (*start='A1'*, *end=None*, *sheet=None*)

Unmerge all cells between the start and end cells. Use defaults to unmerge all cells in the sheet.

Parameters

- **start** (*tuple*, *str*) – Tuple indicating (row, col) or string like 'A1' (default A1)
- **end** (*tuple*, *str*) – Tuple indicating (row, col) or string like 'A1' (default last cell in sheet)
- **sheet** (*str*, *int*, *Worksheet*) – optional, if you want to open or create a different sheet before adding the filter, see [open_sheet](#) (default None)

Returns

Return type None

update_cells (*start*, *end*, *vals*, *sheet=None*, *raw_columns=None*)

Update the values in a given range. The values should be listed in order from left to right across rows.

Parameters

- **start** (*tuple, str*) – tuple indicating (row, col) or string like ‘A1’
- **end** (*tuple, str*) – tuple indicating (row, col) or string like ‘Z20’
- **vals** (*list*) – array of values to populate
- **sheet** (*str, int, Worksheet*) – optional, if you want to open a different sheet first, see `open_sheet` (default None)
- **raw_columns** (*list, int*) – optional, list of column numbers in the google sheet that should be interpreted as “RAW” input

Returns

Return type None

client = None

(*Client*) - Instance of gspread_pandas *Client*

email

(*str*) - E-mail for the currently authenticated user

sheet = None

(*gspread.worksheet.Worksheet*) - Currently open Worksheet

sheets

(*list*) - List of available Worksheets

spread = None

(*gspread.spreadsheet.Spreadsheet*) - Currently open Spreadsheet

url

(*str*) - Url for this spreadsheet

7.1.1.2 gspread_pandas.client module

```
class gspread_pandas.client.Client (user='default', config=None, scope=['openid',
    'https://www.googleapis.com/auth/drive',
    'https://www.googleapis.com/auth/userinfo.email',
    'https://www.googleapis.com/auth/spreadsheets'],
    creds=None, session=None, load_dirs=False)
```

Bases: `gspread.client.Client`

The `gspread_pandas Client` extends `Client` and authenticates using credentials stored in `gspread_pandas config`.

This class also adds a few convenience methods to explore the user’s google drive for spreadsheets.

Parameters

- **user** (*str*) – optional, string indicating the key to a users credentials, which will be stored in a file (by default they will be stored in `~/.config/gspread_pandas/creds/<user>` but can be modified with `creds_dir` property in config). If using a Service Account, this will be ignored. (default “default”)
- **config** (*dict*) – optional, if you want to provide an alternate configuration, see `get_config` (default None)
- **scope** (*list*) – optional, if you’d like to provide your own scope (default `default_scope`)
- **creds** (*google.auth.credentials.Credentials*) – optional, pass credentials if you have those already (default None)

- **session** (*google.auth.transport.requests.AuthorizedSession*) – optional, pass a `google.auth.transport.requests.AuthorizedSession` or a `requests.Session` and `creds` (default `None`)
- **load_dirs** (*bool*) – optional, whether you want to load directories and paths on instantiation. if you refresh directories later or perform an action that requires them, they will be loaded at that time. For speed, this is disabled by default (default `False`)

create_folder (*path, parents=True*)

Create a new folder in your Google drive.

Parameters

- **path** (*str*) – folder path
- **parents** (*bool*) – if `True`, create parent folders as needed (Default value = `True`)

Returns information for the created directory

Return type dict

find_folders (*folder_name_query=""*)

Return all folders that the user has access to containing `folder_name_query` in the name.

Parameters **folder_name_query** (*str*) – Case insensitive string to search in folder name. If empty, it will return all folders.

Returns List of folders. Each folder is a dict with the following keys: `id`, `kind`, `mimeType`, and `name`.

Return type list

find_spreadsheet_files_in_folders (*folder_name_query*)

Return all spreadsheets that the user has access to in all the folders that contain `folder_name_query` in the name. Returns as a dict with each key being the folder name and the value being a list of spreadsheet files.

Parameters **folder_name_query** (*str*) – Case insensitive string to search in folder name

Returns Spreadsheets in each folder. Each entry is a dict with the folder name as the key and a list of spreadsheets as the value. Each spreadsheet is a dict with the following keys: `id`, `kind`, `mimeType`, and `name`.

Return type dict

list_spreadsheet_files (*title=None*)

Return all spreadsheets that the user has access to.

Parameters **title** (*str*) – name of the spreadsheet, if none is passed it'll return every file (default `None`)

Returns List of spreadsheets. Each spreadsheet is a dict with the following keys: `id`, `kind`, `mimeType`, and `name`.

Return type list

list_spreadsheet_files_in_folder (*folder_id*)

Return all spreadsheets that the user has access to in a specific folder.

Parameters **folder_id** (*str*) – ID of a folder, see `find_folders`

Returns List of spreadsheets. Each spreadsheet is a dict with the following keys: `id`, `kind`, `mimeType`, and `name`.

Return type list

login()

Override login since AuthorizedSession now takes care of automatically refreshing tokens when needed.

move_file (*file_id*, *path*, *create=False*)

Move a file to the given path.

Parameters

- **file_id** (*str*) – file id
- **path** (*str*) – folder path. A path starting with / will use your drive, for shared drives the path will start with <Shared Drive Name>/ (no leading /)
- **create** (*bool*) – whether to create any missing folders (Default value = False)

open (*title*)

Opens a spreadsheet.

Parameters **title** (*str*) – 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')
```

refresh_directories()

Refresh list of directories for the current user.

directories

(*list*) - list of dicts for all available directories for the current user

email

(*str*) - E-mail for the currently authenticated user

root

(*dict*) - the info for the top level Drive directory for current user

scope = None

(*list*) - Feeds included for the OAuth2 scope

7.1.1.3 gspread_pandas.conf module

`gspread_pandas.conf.get_config` (*conf_dir=None*, *file_name='google_secret.json'*)

Get config for Google client. Looks in `~/config/gspread_pandas/google_secret.json` by default but you can override it with `conf_dir` and `file_name`. The `creds_dir` value will be set to `conf_dir/creds` and the directory will be created if it doesn't exist; if you'd like to override that you can do so by changing the 'creds_dir' value in the dict returned by this function.

Download json from <https://console.developers.google.com/apis/credentials>

Parameters

- **conf_dir** (*str*) – Full path to config dir (Default value = `get_config_dir()`)
- **file_name** (*str*) – (Default value = "google_secret.json")

Returns Dict with necessary contents of `google_secret.json`

Return type dict

```
gsread_pandas.conf.get_creds (user='default', config=None, scope=['openid',  
                           'https://www.googleapis.com/auth/drive',  
                           'https://www.googleapis.com/auth/userinfo.email',  
                           'https://www.googleapis.com/auth/spreadsheets'],  
                           creds_dir=None, save=True)
```

Get google google.auth.credentials.Credentials for the given user. If the user doesn't have previous creds, they will go through the OAuth flow to get new credentials which will be saved for later use. Credentials will be saved in config['creds_dir'], if this value is not set, then they will be stored in a folder named `creds` in the default config dir (either `~/config/gspread_pandas` or

`$GSPREAD_PANDAS_CONFIG_DIR`)

Alternatively, it will get credentials from a service account.

Parameters

- **user** (*str*) – Unique key indicating user's credentials. This is not necessary when using a ServiceAccount and will be ignored (Default value = "default")
- **config** (*dict*) – Optional, dict with "client_id", "client_secret", and "redirect_uris" keys for OAuth or "type", "client_email", "private_key", "private_key_id", and "client_id" for a Service Account. If None is passed, it will call `get_config()` (Default value = None)
- **creds_dir** (*str, Path*) – Optional, directory to load and store creds from/in. If None, it will use the `creds` subdirectory in the default config location. (Default value = None)
- **scope** (*list*) – Optional, scope to use for Google Auth (Default value = `default_scope`)

Returns Google credentials that can be used with `gspread`

Return type `google.auth.credentials.Credentials`

Code should be run through `black`, `isort`, and `flake8` before being merged. Pre-commit takes care of it for you, but you need to have Python 3 installed to be able to run `black`. To contribute, please fork the repo, create a feature branch, push it to your repo, then create a pull request.

To install and set up the environment after you fork it (replace *aiguofer* with your username):

```
$ git clone https://github.com/aiguofer/gspread-pandas.git && cd gspread-pandas
$ pip install -e ".[dev]"
$ pre-commit install
```

8.1 Testing

Our tests leverage `betamax` to remember HTTP interactions with the API. In order to add new tests that change the requests to `betamax`, you'll need to have Service Account credentials stored as `google_secret.json` in the root project directory. You can then re-record tests by deleting the necessary cassettes in `tests/cassettes` then running:

```
$ GSPREAD_RECORD=true pytest <path_to_test>
```

NOTE: Currently, the tests don't do any setup and teardown of expected directories/files in the Google Drive. My main concern in implementing this is that somehow it might mistakenly use a specific user's credentials and delete important stuff. If you have any ideas here I'd be happy to discuss.

8.2 Versions

In order to bump versions, we use `bumpversion`. This will take care of adding an entry in the CHANGELOG for the new version and bumping the version everywhere it needs to. This will also create a git tag for the specific version.

8.3 CI

CI is managed by Github Actions: python-package.yml - workflow for testing and linting for each python version on every push tagged-release.yml - workflow that does ^ and then creates a github release only on tagged push python-publish.yml - workflow that publishes package to PyPi when a github release is created

All notable changes to this project will be documented in this file.

The format is based on [Keep a Changelog](#) and this project adheres to [Semantic Versioning](#).

9.1 [Unreleased]

9.2 [3.2.2] - 2022-06-29

9.3 Changed

- Update CI and its documentation

9.4 [3.2.1] - 2022-06-29

9.5 Fixed

- Don't use the deprecated pandas option *inplace = True*
- Fix hanging auth test
- Example in docs

9.6 Removed

- Fix for early pandas 1.0 is no longer necessary

9.7 [3.2.0] - 2022-03-19

9.8 [3.1.0] - 2022-03-19

9.9 Fixed

- OOB auth flow was removed from google-auth, [see here](#). Fixed it by running local server to listen for token.

9.10 [3.0.4] - 2022-01-18

9.11 Fixed

- Frozen column field in `gridProperties` changed from `frozenColCount` to `frozenColumnCount`

9.12 [3.0.3] - 2022-01-04

9.13 Fixed

- Potential issue with resizing in `clear_sheet`

9.14 Added

- Section about resizing issues to README

9.15 [3.0.2] - 2021-12-29

9.16 Removed

- Remove deprecated param `raw_column_names`

9.17 [3.0.1] - 2021-12-28

9.18 Changed

- Pipeline changes

9.19 [3.0.0] - 2021-12-28

9.20 Changed

- Upgrade gsread to ≥ 5

9.21 Removed

- BREAKING: Removed Python 2.7 support
- Removed Travis pipeline

9.22 Added

- CI pipeline using Github Actions

9.23 [2.3.1] - 2021-11-29

9.24 Fixed

- Fix gsread 5 incompatibility by blocking requirements

9.25 [2.3.0] - 2021-03-21

9.26 Changed

- Switch to pycryptodome instead of pycrypto to fix tests
- Switch from future to six for python 2 compat

9.27 Fixed

- Fix docstring about default sheet param for *open*
- Fix numpy deprecation warning about numeric-style types
- Fix google quota exceed retry

9.28 Added

- Added ability to use shared drives

9.29 [2.2.4] - 2021-01-24

9.30 Fixed

- Fix dtype issues for numeric+null values for pandas 1.0
- Fix auth within Colab

9.31 [2.2.3] - 2020-03-26

9.32 Fixed

- Ensure we respect XDG_CONFIG_HOME
- Use current dir if no \$HOME env var is set (in some cloud environments and Colab)

9.33 [2.2.2] - 2020-03-21

9.34 Changed

- Use Path for various paths in conf.py

9.35 [2.2.1] - 2020-01-05

9.36 Added

- Added a section explaining the main concepts of Gspread-Pandas to the docs.

9.37 Fixed

- Fixed the auto-retry when encountering the 100 second quota limit (thanks @rodelrod).
- All `Client` objects should now be monkeypatched to retry on 100 s quota error. Before it was only those within a `Spread` instance.

9.38 Changed

- When a `Spread` is opened, it will now open the first worksheet by default.

9.39 [2.2.0] - 2019-11-17

9.40 Added

- Added `formula_columns` and `unformatted_columns` as options to `Spread.sheet_to_df` to allow usage of `ValueRenderOption` for specific columns.
- Added test for python 3.7 and 3.8 in travis

9.41 Fixed

- Bug with certain functions when `index=False`
- Bug when all columns are `raw_columns`
- Fixed travis tests

9.42 Deprecated

- `raw_column_names` is deprecated in favor of `raw_columns`. It will be removed in v3.

9.43 Changed

- bumpversion config now in `setup.cfg`
- Docstrings now use `docformatter` format

9.44 [2.1.3] - 2019-08-25

9.45 Added

- Added `Spread.add_permission` to differentiate adding a single permission vs adding multiple permissions.

9.46 Changed

- Ensure license is packaged with `sdist`
- Unified and simplified project and test configuration settings

9.47 [2.1.2] - 2019-07-10

9.48 Added

- Added test for `open_sheet`

9.49 Fixed

- Client directories property now works as expected
- Various fixes for tests

9.50 Changed

- Tox must run correctly before being able to push a new tag

9.51 [2.1.1] - 2019-07-10

9.52 Fixed

- Fixed import in `Spread` after refactor, oops! should have ran tests

9.53 [2.1.0] - 2019-07-10

9.54 Added

- Client now has an optional `load_dirs` param which default to `False`

9.55 Changed

- **BREAKING:** Refactored `Spread` into its own file. If you were importing like `from gsread_pandas.client import Spread` you will need to change to `from gsread_pandas.spread import Spread`.
- Directories and paths are no longer loaded by default. However, if you try to use any functionality that requires it, it'll load it at that point.

9.56 Fixed

- If a file doesn't have `parents` it'll no longer break (thanks @shredding) (#29)
- `$XDG_CONFIG_HOME` should now be respected

- If you don't have Drive API access in the scope, it should now still work and print a message instead

9.57 [2.0.0] - 2019-06-12

9.58 Added

- Test python 3.7, Windows, and MacOS
- You can now iterate over worksheets like: `for sheet in spread`
- `Spread.df_to_sheet` can now flatten multi-level headers using the `flatten_headers_sep` param
- Add ability to set permissions on spreadsheets
- Add ability to create and move folders and spreadsheets
- A session can now be passed directly to a `Client`
- A `raw_column_names` param to `Spread.df_to_sheet` to force specific columns to be sent to the Google Sheets API as RAW input so it doesn't get interpreted as a number, date, etc.

9.59 Removed

- BREAKING: Removed `start/end_row/col` params from `add_filter`
- BREAKING: Removed `user_creds_or_client` param from `Spread`
- BREAKING: Removed `user_or_creds` param from `Client`

9.60 Changed

- The `credentials` param is now called `creds` everywhere
- Test suite is now a lot more robust
- Use `google-auth` instead of the now deprecated `oauth2client` library. This moves the retry code into that library.
- Default config will now be in `C:\Users\\AppData\gspread_pandas` on Windows

9.61 Fixed

- Things should now work when passing a `Worksheet` object to `Spread.open`

9.62 [1.3.1] - 2019-05-17

9.62.1 Fixed

- Passing 0 to `sheet`` param in any function should work now
- When using multi-row column headers in a spreadsheet, the index name should now be better identified

- Spread;update_cells should now work when passing a single cell value
- When start != 'A1', freeze_rows/headers should now correctly freeze the right amount of rows/headers so the index and columns are frozen

9.63 [1.3.0] - 2019-04-30

9.63.1 Added

- Function to merge_cells
- Function to unmerge_cells
- Option to merge_headers in df_to_sheet
- Retry when exceeding the 100s quota

9.63.2 Fixed

- Fix passing 0 for freeze_index or freeze_headers. This essentially “unfreezes”
- When the index has no name and you have a multi-level header, it will no longer fill in “index” as the index header

9.63.3 Deprecated

- Spread will no longer use the ‘user_creds_or_client’ param in v2. Instead, it will have optional ‘credentials’, ‘client’, and ‘user’ params. If creds or a client are passed, the user will be ignored. Otherwise, it’ll use the user, which will default to “default”
- Client will no longer use the ‘user_or_creds’ param in v2. Instead, it will have optional ‘credentials’ and ‘user’ params. If creds passed, the user will be ignored. Otherwise, it’ll use the user, which will default to “default”
- Spread.add_filter will be standardized to use ‘start’ and ‘end’ like other functions and the start/end_row/col are deprecated and will be removed in v2

9.63.4 Changed

- Exceptions are no longer raised while handling another exception. This should prevent the “During handling of the above exception, another exception occurred” message
- When opening a new Spreadsheet, the SpreadsheetNotFound exception will no longer be a “catchall” for any errors. If an error other than actually not finding the Spreadsheet occurs, it’ll be raised.
- Default value for the user param in util.get_config was changed to “default”

9.64 [1.2.2] - 2019-04-15

9.64.1 Fixed

- Fix passing only one of freeze_index or freeze_headers = True

9.65 [1.2.1] - 2018-08-30

9.65.1 Fixed

- Fixed `__version__` string for bumpversion using black

9.66 [1.2.0] - 2018-08-30

9.66.1 Added

- Add config files and pre-commit hooks for isort, black, and flake8
- Add config files for isort, black, and flake8

9.66.2 Fixed

- Fixed `clear_sheet` when there are frozen rows/cols
- Small fixes in README

9.66.3 Changed

- Changed from reST docstrings to numpy docstrings
- Updated README to include more in contributing section

9.67 [1.1.3] - 2018-07-07

9.67.1 Added

- Added unit tests for util

9.67.2 Fixed

- Fix `parse_df_col_names` when df has a multi-index
- Fix `parse_sheet_index` when using last column as index
- Fix `fillna` when using categorical variables

9.68 [1.1.2] - 2018-06-23

9.68.1 Fixed

- Fix issue with basestring usage

9.68.2 Changed

- Remove Python 3.4 from travis tests

9.69 [1.1.1] - 2018-06-13

9.69.1 Changed

- `Spread.clear_sheet` now doesn't resize to 0 since V4 is much more efficient at making batch updates. This should help prevent formulas that point to these sheets from breaking.

9.70 [1.1.0] - 2018-06-02

9.70.1 Fixed

- Now works with gspread 3.0
- `Spread.freeze` is working again

9.70.2 Changed

- Moved a lot of the credential handling into functions in `gsread_pandas.conf`
- New `get_creds` function allows you to get `OAuth2Credentials` and pass them in to a `Client` or `Spread`
- Some functions were moved to `gsread_pandas.util`

9.70.3 Added

- New function `Spread.add_filter` created so that you can add filters to worksheets
- New param `add_filter` added to `Spread.df_to_sheet` to add a filter to uploaded data

9.71 [1.0.5] - 2018-04-14

9.71.1 Fixed

- Added limit to gspread version since 3.0 broke gspread-pandas

9.72 [1.0.4] - 2018-04-08

9.72.1 Fixed

- Change `ValueInputOption` to `USER_ENTERED` so dates and numbers are parsed correctly in Google Sheets

9.73 [1.0.3] - 2018-04-02

9.73.1 Added

- Basic initial test

9.74 [1.0.2] - 2018-04-02

9.74.1 Changed

- Some dependency changes
- Travis deploy will only happen on python 3.6
- Changes to reduce number of `fetch_sheet_metadata` calls

9.75 [1.0.1] - 2018-03-26

9.75.1 Changed

- Replace `pypi-publisher` with `twine` in dev reqs
- Change download url, now it should match the tags from `bumpversion`

9.76 [1.0.0] - 2018-03-26

9.76.1 Added

- There is now a separate `Client` class that extends the `gsread v4 Client` class and adds some functionality. This includes a monkeypatche and hacky workarounds for `gsread 2.0` issues. Once they get fixed upstream I need to remove these.

9.76.2 Changed

- Now supports `gsread 2.0` which uses `Spreadsheets V4 API`, this provides much better performance and reliability. Some APIs might have changed.
- No longer need to chunk update requests, and range requests can use larger chunks
- Some code improvements enabled by `gsread 2.0`
- Removed deprecated params and functions

9.77 [0.16.1] - 2018-03-24

9.77.1 Fixed

- Set up correct credentials for travis pypi push

9.78 [0.16.0] - 2018-03-24

9.78.1 Added

- Test on multiple versions using tox
- Enable travis-ci

9.78.2 Fixed

- Remove dir accidentally pushed by build

9.78.3 Changed

- Moved dev requirements into requirements_dev.txt
- Now using bumpversion for version management
- Minor updates to README
- Documentation now at Read The Docs
- Minor code changes to please flake8
- Deleted update_pypi.sh as releases are now handled by travis

9.79 [0.15.6] - 2018-03-12

9.79.1 Fixed

- Remove code accidentally pushed by build

9.80 [0.15.5] - 2018-03-12

9.80.1 Fixed

- Added dependency version limit for gsread; will remove in next version

9.81 [0.15.4] - 2018-02-13

9.81.1 Fixed

- README example now points to the correct URL (thanks @lionel)
- Calling `parse_sheet_headers` on an empty sheet doesn't break anymore (thanks @taewookim)

9.81.2 Added

- You can now use service account credentials in the config (thanks @marcojetson)

9.82 [0.15.3] - 2017-11-21

9.82.1 Changed

- Always return an Index object from `parse_sheet_headers`

9.83 [0.15.2] - 2017-11-18

9.83.1 Fixed

- Fix `sheet_to_df` when headers are present with no data

9.83.2 Changed

- Minimum Pandas version .20 now required

9.84 [0.15.1] - 2017-10-05

9.84.1 Fixed

- When there are merged cells outside the data range, an exception is no longer thrown.
- Cast `keys()` to a list to fix Python 3 compat

9.85 [0.15.0] - 2017-09-11

9.85.1 Changed

- Added `fill_value` option to `df_to_sheet`

9.85.2 Fixed

- Different application type credentials can be used now
- Some safeguards to prevent certain exceptions
- `df_to_sheet` won't fail when categorical columns have nulls

9.86 [0.14.3] - 2017-06-22

9.86.1 Changed

- Force gsread sheets refresh when refreshing sheets
- Worksheet object can now be passed it to most functions with `sheet` param

9.87 [0.14.2] - 2017-06-18

9.87.1 Added

- Added `url` property for easy linking

9.87.2 Fixed

- Fixed retry for `_retry_get_all_values`

9.88 [0.14.1] - 2017-06-05

9.89 Changed

- Ensure sheet metadata is refreshed after sheet changing activities through use of a decorator
- Retry when calling `get_all_values`
- More robust way to get index when a new sheet is created

9.90 [0.14.0] - 2017-05-25

9.90.1 Added

- Added function to freeze rows/columns to `Spread`
- Added `freeze_index` and `freeze_headers` flags to `df_to_sheet`

9.90.2 Changed

- Don't re-size again when using `replace=True`
- Switch away from deprecated `gspread` functions
- Make functions in `util` non-private

9.90.3 Fixed

- Prevent error when `index > number of columns` in `sheet_to_df`

9.91 [0.13.0] - 2017-04-28

9.91.1 Added

- Added `create_spread` and `create_sheet` params for `Spread` class. This enables creating a spreadsheet or a worksheet during opening. This will require re-authenticating in order to use it

9.92 [0.12.1] - 2017-04-25

9.92.1 Changed

- If using multi-level headings, heading will be shifted up so the top level is not a blank string
- Some functions that don't depend on `self` were moved into `util.py`
- The `headers` param in `sheet_to_df` was deprecated in favor of `header_rows`

9.92.2 Fixed

- I introduced some small bugs with the v4 api changes when a sheet is not found, they now work as expected even when a new sheet is created
- The list of sheets is now refreshed when one is deleted

9.93 [0.12.0] - 2017-03-31

9.93.1 Added

- Add Sheets API v4 client to `self.clientv4`

9.93.2 Fixed

- Merged cells now all get the right value in `sheet_to_df`
- You can now pass `replace=True` when a sheet has frozen rows/cols

9.94 [0.11.2] - 2017-03-22

9.94.1 Changed

- Minor change to README

9.95 [0.11.1] - 2017-03-22

9.95.1 Added

- Added note about `EOFError` when verifying Oauth in Rodeo

9.95.2 Changed

- Add retry method for `sheet.range` to work around 'Connection Broken' error

9.95.3 Fixed

- Fixed clearing only rows with `clear_sheet`

9.96 [0.11.0] - 2017-02-14

9.96.1 Changed

- Only clear up to first row in `clear_sheet` so that data filters will persist
- Moved default config from `~/ .google/` to `~/ .config/gspread_pandas`

9.96.2 Fixed

- Allow passing index 0 to `open`
- Fixed changelog

9.97 [0.10.1] - 2017-01-26

9.97.1 Added

- Added troubleshooting for `certifi` issue in README

9.97.2 Changed

- Only catch `SpreadsheetNotFound` exceptions when opening a spreadsheet

9.98 [0.10.0] - 2017-01-18

9.98.1 Added

- Added optional `create` param to `open_sheet` to create it if it doesn't exist
- Added optional `start` param to `df_to_sheet`, will take tuple or address as str

9.98.2 Changed

- Improved docs, changed to `rst`
- Made some variables private
- Improved `__str__` output
- Switch to using exceptions from `gsread`
- `spread` param is now required for `open`
- When current sheet is deleted, `self.sheet` is set to `None`
- Improved versioning, switched to [Semantic Versioning](#)

9.98.3 Fixed

- Fixed chunk calculation in Python 3
- Sheet names are case insensitive, fixed `find_sheet`

9.98.4 Deprecated

- Deprecate `open_or_create_sheet` function in favor of `create=True` param for `open_sheet`
- Deprecate `start_row` and `start_col` in `df_to_sheet` in favor of `start` param

9.99 [0.9] - 2016-12-07

9.99.1 Added

- Add `__repr__` and `__str__` to show the active
- Add user's email as a property to `Spread`. I recommend deleting existing Oauth credentials and re-creating them with new permissions
- Allow importing with: `from gsread_pandas import Spread`
- Added `CHANGELOG.md`

9.99.2 Changed

- Restrict scope to only necessary endpoints
- Add retry for updating cells in case an error occurs
- Minor changes to README.md

9.99.3 Fixed

- Fixed the use of `start_row > 1`

9.100 [0.8] - 2016-11-11

9.100.1 Added

- Add python 3 build to `update_pypi.sh` script

9.100.2 Fixed

- Oauth flow now uses correct properties

9.101 [0.7] - 2016-11-10

9.101.1 Changed

- Made python 3 compatible using `future`

9.102 [0.6] - 2016-10-27

9.102.1 Changed

- Change defaults in `sheet_to_df` to include index and header
- Raise error when missing google client config file

9.103 [0.5] - 2016-10-19

9.103.1 Changed

- Improve decorators more using `decorator.decorator`

9.104 [0.4] - 2016-10-19

9.104.1 Added

- Pypi update script

9.105 Changed

- Improve decorators using `functools.wraps`

9.106 [0.3] - 2016-10-19

9.106.1 Changed

- Add `ensure_auth` decorator to most functions to re-auth if needed
- Chunk requests to prevent timeouts
- Improved `clear_sheet` by resizing instead of deleting and re-creating

9.107 [0.2] - 2016-10-12

9.107.1 Added

- Code migrated
- Example usage in README
- Add requirements

9.108 [0.1] - 2016-10-11

9.108.1 Added

- README
- initial code migrated

CHAPTER 10

Indices and tables

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

g

`gsread_pandas.client`, 21
`gsread_pandas.conf`, 23
`gsread_pandas.spread`, 15

A

`add_filter()` (*gsread_pandas.spread.Spread method*), 16
`add_permission()` (*gsread_pandas.spread.Spread method*), 16
`add_permissions()` (*gsread_pandas.spread.Spread method*), 17

C

`clear_sheet()` (*gsread_pandas.spread.Spread method*), 17
`Client` (*class in gsread_pandas.client*), 21
`client` (*gsread_pandas.spread.Spread attribute*), 21
`create_folder()` (*gsread_pandas.client.Client method*), 22
`create_sheet()` (*gsread_pandas.spread.Spread method*), 17

D

`delete_sheet()` (*gsread_pandas.spread.Spread method*), 17
`df_to_sheet()` (*gsread_pandas.spread.Spread method*), 17
`directories` (*gsread_pandas.client.Client attribute*), 23

E

`email` (*gsread_pandas.client.Client attribute*), 23
`email` (*gsread_pandas.spread.Spread attribute*), 21

F

`find_folders()` (*gsread_pandas.client.Client method*), 22
`find_sheet()` (*gsread_pandas.spread.Spread method*), 18
`find_spreadsheet_files_in_folders()` (*gsread_pandas.client.Client method*), 22
`freeze()` (*gsread_pandas.spread.Spread method*), 18

G

`get_config()` (*in module gsread_pandas.conf*), 23
`get_creds()` (*in module gsread_pandas.conf*), 23
`get_sheet_dims()` (*gsread_pandas.spread.Spread method*), 18
`gsread_pandas.client` (*module*), 21
`gsread_pandas.conf` (*module*), 23
`gsread_pandas.spread` (*module*), 15

L

`list_permissions()` (*gsread_pandas.spread.Spread method*), 18
`list_spreadsheet_files()` (*gsread_pandas.client.Client method*), 22
`list_spreadsheet_files_in_folder()` (*gsread_pandas.client.Client method*), 22
`login()` (*gsread_pandas.client.Client method*), 22

M

`merge_cells()` (*gsread_pandas.spread.Spread method*), 19
`move()` (*gsread_pandas.spread.Spread method*), 19
`move_file()` (*gsread_pandas.client.Client method*), 23

O

`open()` (*gsread_pandas.client.Client method*), 23
`open()` (*gsread_pandas.spread.Spread method*), 19
`open_sheet()` (*gsread_pandas.spread.Spread method*), 19
`open_spread()` (*gsread_pandas.spread.Spread method*), 20

R

`refresh_directories()` (*gsread_pandas.client.Client method*), 23
`refresh_spread_metadata()` (*gsread_pandas.spread.Spread method*), 20

`root` (*gsread_pandas.client.Client* attribute), 23

S

`scope` (*gsread_pandas.client.Client* attribute), 23

`sheet` (*gsread_pandas.spread.Spread* attribute), 21

`sheet_to_df()` (*gsread_pandas.spread.Spread* method), 20

`sheets` (*gsread_pandas.spread.Spread* attribute), 21

`Spread` (class in *gsread_pandas.spread*), 15

`spread` (*gsread_pandas.spread.Spread* attribute), 21

U

`unmerge_cells()` (*gsread_pandas.spread.Spread* method), 20

`update_cells()` (*gsread_pandas.spread.Spread* method), 20

`url` (*gsread_pandas.spread.Spread* attribute), 21