gspread-pandas Documentation

Release 1.3.1

Diego Fernandez

Jun 12, 2019

Contents

author: Diego Fernandez

Links:

- Documentation
- Source code
- Short video tutorial

CHAPTER 1

Overview

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 gspread in the backend for most of the heavylifting, 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.

Some key goals/features:

- Nicely handle headers and indexes
- 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 handling of merged cells when pulling data
- Nicely handle large data sets and auto-retries
- Enable creation of filters
- Handle retries when exceeding 100s quota
- When pushing DataFrames with MultiIndex columns, allow merging or flattening headers
- Ability to nicely handle Spreadsheet permissions

CHAPTER 2

Installation / Usage

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 **Other** 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

```
from __future__ import print_function
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', 'A1', ['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 instanciate a Client separately and query folders and
# instanciate other Spread objects by passing in the Client
client = Client()
# Assumming 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))
```

CHAPTER 4

Troubleshooting

4.1 SSL Error

If you're getting an SSL related error or can't seem to be able to open existing spreadsheets that you have access to, you might be running into an issue caused by `certifi`. This has mainly been experienced on RHEL and CentOS running Python 2.7. You can read more about it in issue 223 and issue 354 but, in short, the solution is to either install a specific version of `certifi` that works for you, or remove it altogether.

```
pip install certifi==2015.4.28
```

or

```
pip uninstall certifi
```

4.2 EOFError in Rodeo

If you're trying to use `gspread_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 gspread_pandas import Spread; Spread('<user_key>','')"
```

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

CHAPTER 5

Configuration

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 withSpreadsheets 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 berefreshed 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 instanciated, 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 `gspread_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 `gspread_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.

CHAPTER 6

Modules

6.1 gspread_pandas

6.1.1 Submodules

6.1.1.1 gspread_pandas.client module

class gspread_pandas.client.**Spread**(*spread, sheet=None, 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, permissions=None*)

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 None)
- **config** (*dict*) – optional, if you want to provide an alternate configuration, see *get_config* (default None)
- **create_sheet** (*bool*) – whether to create the spreadsheet if it doesn't exist, it wil use the `spread` value as the sheet title (default False)

- **create_spread** (*bool*) – whether to create the sheet if it doesn't exist, it wil 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 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")
- **creds** (*google.auth.credentials.Credentials*) – optional, pass credentials if you have those already (default None)
- **client** (*Client*) – optionall, if you've already instanciated 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_permissions(*permissions*)

Add permissions to the current spreadsheet.

The format of each string 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 ommited, '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** (*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_column_names=[]*, *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_column_names** (*list, str*) – optional, list of columns from your dataframe that you want interpreted as RAW input in google sheets
- **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 spreadsheet if it doesn't exist, it wil use the `spread` value as the sheet title (default False)
- **create_spread** (*bool*) – whether to create the sheet if it doesn't exist, it wil 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 wil 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*, *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)
- **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=[]*)
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 indexes 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.models.Worksheet) - Currently open Worksheet

sheets
(list) - List of available Worksheets

spread = None
(gspread.models.Spreadsheet) - Currently open Spreadsheet

url
(str) - Url for this spreadsheet

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*)

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)

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 sepcific 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
- **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 avaliable 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

6.1.1.2 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

gspread_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*) – 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

CHAPTER 7

Contributing

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
```

7.1 Testing

Our tests levarage 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 cassetes 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.

7.2 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.

7.3 CI

Tests will run on Travis CI using Tox, and they'll test on a variety of Python versions on Linux, and test only latest Python on Mac and Windows. If a version tag is pushed and the tests pass, the new version will be pushed to PyPi by Travis.

CHAPTER 8

Change Log

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.

8.1 [Unreleased]

8.2 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`

8.3 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

8.4 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\<user>\AppData\gspread_pandas` on Windows

8.5 Fixed

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

8.6 [1.3.1] - 2019-05-17

8.6.1 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

8.7 [1.3.0] - 2019-04-30

8.7.1 Added

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

8.7.2 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

8.7.3 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

8.7.4 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"

8.8 [1.2.2] - 2019-04-15

8.8.1 Fixed

- Fix passing only one of freeze_index or freeze_headers = True

8.9 [1.2.1] - 2018-08-30

8.9.1 Fixed

- Fixed __version__ string for bumpversion using black

8.10 [1.2.0] - 2018-08-30

8.10.1 Added

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

8.10.2 Fixed

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

8.10.3 Changed

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

8.11 [1.1.3] - 2018-07-07

8.11.1 Added

- Added unit tests for util

8.11.2 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

8.12 [1.1.2] - 2018-06-23

8.12.1 Fixed

- Fix issue with basestring usage

8.12.2 Changed

- Remove Python 3.4 from travis tests

8.13 [1.1.1] - 2018-06-13

8.13.1 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.

8.14 [1.1.0] - 2018-06-02

8.14.1 Fixed

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

8.14.2 Changed

- Moved a lot of the credential handling into functions in gspread_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 `gspread_pandas.util`

8.14.3 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

8.15 [1.0.5] - 2018-04-14

8.15.1 Fixed

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

8.16 [1.0.4] - 2018-04-08

8.16.1 Fixed

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

8.17 [1.0.3] - 2018-04-02

8.17.1 Added

- Basic initial test

8.18 [1.0.2] - 2018-04-02

8.18.1 Changed

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

8.19 [1.0.1] - 2018-03-26

8.19.1 Changed

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

8.20 [1.0.0] - 2018-03-26

8.20.1 Added

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

8.20.2 Changed

- Now supports gspread 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 gspread 2.0
- Removed deprecated params and functions

8.21 [0.16.1] - 2018-03-24

8.21.1 Fixed

- Set up correct credentials for travis pypi push

8.22 [0.16.0] - 2018-03-24

8.22.1 Added

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

8.22.2 Fixed

- Remove dir accidentally pushed by build

8.22.3 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

8.23 [0.15.6] - 2018-03-12

8.23.1 Fixed

- Remove code accidentally pushed by build

8.24 [0.15.5] - 2018-03-12

8.24.1 Fixed

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

8.25 [0.15.4] - 2018-02-13

8.25.1 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)

8.25.2 Added

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

8.26 [0.15.3] - 2017-11-21

8.26.1 Changed

- Always return an Index object from parse_sheet_headers

8.27 [0.15.2] - 2017-11-18

8.27.1 Fixed

- Fix sheet_to_df when headers are present with no data

8.27.2 Changed

- Minimum Pandas version .20 now required

8.28 [0.15.1] - 2017-10-05

8.28.1 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

8.29 [0.15.0] - 2017-09-11

8.29.1 Changed

- Added `fill_value` option to df_to_sheet

8.29.2 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

8.30 [0.14.3] - 2017-06-22

8.30.1 Changed

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

8.31 [0.14.2] - 2017-06-18

8.31.1 Added

- Added `url` property for easy linking

8.31.2 Fixed

- Fixed retry for _retry_get_all_values

8.32 [0.14.1] - 2017-06-05

8.33 Changed

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

8.34 [0.14.0] - 2017-05-25

8.34.1 Added

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

8.34.2 Changed

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

8.34.3 Fixed

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

8.35 [0.13.0] - 2017-04-28

8.35.1 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

8.36 [0.12.1] - 2017-04-25

8.36.1 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`

8.36.2 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

8.37 [0.12.0] - 2017-03-31

8.37.1 Added

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

8.37.2 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

8.38 [0.11.2] - 2017-03-22

8.38.1 Changed

- Minor change to README

8.39 [0.11.1] - 2017-03-22

8.39.1 Added

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

8.39.2 Changed

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

8.39.3 Fixed

- Fixed clearing only rows with `clear_sheet`

8.40 [0.11.0] - 2017-02-14

8.40.1 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`

8.40.2 Fixed

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

8.41 [0.10.1] - 2017-01-26

8.41.1 Added

- Added troubleshooting for `certifi` issue in `README`

8.41.2 Changed

- Only catch `SpreadsheetNotFound` exceptions when opening a spreadsheet

8.42 [0.10.0] - 2017-01-18

8.42.1 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

8.42.2 Changed

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

8.42.3 Fixed

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

8.42.4 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

8.43 [0.9] - 2016-12-07

8.43.1 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 gspread_pandas import Spread`
- Added `CHANGELOG.md`

8.43.2 Changed

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

8.43.3 Fixed

- Fixed the use of `start_row` > 1

8.44 [0.8] - 2016-11-11

8.44.1 Added

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

8.44.2 Fixed

- Oauth flow now uses correct properties

8.45 [0.7] - 2016-11-10

8.45.1 Changed

- Made python 3 compatible using future

8.46 [0.6] - 2016-10-27

8.46.1 Changed

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

8.47 [0.5] - 2016-10-19

8.47.1 Changed

- Improve decorators more using `decorator.decorator`

8.48 [0.4] - 2016-10-19

8.48.1 Added

- Pypi update script

8.49 Changed

- Improve decorators using `functools.wraps`

8.50 [0.3] - 2016-10-19

8.50.1 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

8.51 [0.2] - 2016-10-12

8.51.1 Added

- Code migrated
- Example usage in README
- Add requirements

8.52 [0.1] - 2016-10-11

8.52.1 Added

- README
- initial code migrated

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index

S

U