datawrapper Documentation

Release 0.4.3

Sergio Sanchez

Jun 24, 2020

Contents:

This project provides a light-weight Python wrapper for the Datawrapper API (v3), developed by Sergio Sanchez. While it is not developed by Datawrapper officially, you can use it with your API credentials from datawrapper.de

CHAPTER 1

datawrapper

This project provides a light-weight Python wrapper for the Datawrapper API (v3), developed by Sergio Sanchez. While it is not developed by Datawrapper officially, you can use it with your API credentials from datawrapper.de

See https://developer.datawrapper.de/docs/getting-started.

- Free software: MIT license
- Documentation: https://datawrapper.readthedocs.io.

1.1 Features

- Retrieve your account information (including folders).
- Add data to charts, tables or maps.
- Create charts, tables or maps - and add data from a *pandas.DataFrame* in one call!
- Update chart descriptions.
- Publish charts, tables or maps.
- Retrieve chart properties, update its metadata, and other information.
- Display a chart (as output of notebook cell - it gets weird because interactivity ¯_()_/¯)
- Retrieve a chart, table or map's iframe code to embed.
- Export chart as png (still working on the svg and pdf parts).

- Move charts across folders and organizations.
- Delete charts.
- Get a list of all your charts.

1.2 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

CHAPTER 2

Installation

2.1 Stable release

To install datawrapper, run this command in your terminal:

```
$ pip install datawrapper
```

This is the preferred method to install datawrapper, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for datawrapper can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/chekos/datawrapper
```

Or download the tarball:

```
$ curl -OJL https://github.com/chekos/datawrapper/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

Before using *datawrapper* in a project you'll need to get an access token from Datawrapper. See https://app.datawrapper.de/account/api-tokens

To use datawrapper in a project:

```
from datawrapper import Datawrapper

dw = Datawrapper(access_token = "INSERT_YOUR_ACCESS_TOKEN_HERE")
```

Now you have access to your Datawrapper account.

To view your account info run:: dw.account_info()

Create a chart new_chart_info = dw.create_chart(title = 'New chart!', chart_type = 'd3-bars-stacked')

This returns a JSON with your new chart's info, including its ID which you will need to update, add data to, move, display or delete said chart.

You could also pass a pandas.DataFrame as data in the same call and you'll have created a chart and uploaded the data at once.

```
df = pd.read_csv("https://raw.githubusercontent.com/chekos/datasets/master/data/
↪datawrapper_example.csv", sep=';')
new_chart_info = dw.create_chart(title = 'New chart 2!', chart_type = 'd3-bars-stacked
↪', data  = df)
new_chart_info

>>>> {'title': 'New chart 2!',
        'theme': 'default',
        'type': 'd3-bars-stacked',
        'language': 'en-US',
        'metadata': {'data': {}},
        'authorId': 163125,
        'id': 'OsgIU',
        'lastModifiedAt': '2019-12-11T23:17:35.826Z',
```

(continues on next page)

(continued from previous page)

```
        'createdAt': '2019-12-11T23:17:35.826Z',
        'url': '/v3/charts/OsgIU'}
```

Your chart will have a different 'id'. That is super important because that's how you edit your chart!

Update the chart's description:

```
dw.update_description(
    chart_id = new_chart_info['id'],
    source_name = 'UN Population Division',
    source_url = 'https://population.un.org/wup/',
    byline = 'Your name here!',
)
>>>> Chart updated!
```

For others to see your marvelous creation you need to publish your chart!

```
dw.publish_chart(chart_id = new_chart_info['id'])
```

By default, *datawrapper* (the python package) will attempt to display your chart as the cell's output. This feature is still being tested so it might be changed to not do this by default in the future.

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/chekos/datawrapper/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

datawrapper could always use more documentation, whether as part of the official datawrapper docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/chekos/datawrapper/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *datawrapper* for local development.

1. Fork the *datawrapper* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/datawrapper.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv datawrapper
$ cd datawrapper/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 datawrapper tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.org/chekos/datawrapper/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_datawrapper
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER 5

Credits

5.1 Development Lead

- Sergio Sanchez <chekos@tacosdedatos.com>

5.2 Contributors

@alex-pobeditel-2004

CHAPTER 6

History

6.1 0.4.3 (2020-04-10)

- Encode data in UTF-8 by default. Thanks to @alex-pobeditel-2004

6.2 0.4.2 (2019-12-17)

- Bug fixes

6.3 0.4.1 (2019-12-17)

- Added borderWidth param to .export_chart()

6.4 0.4.0 (2019-12-17)

- Added ability to export charts.

6.5 0.3.0 (2019-12-12)

- Added functionality. *get_charts()*, *get_folders()*, *update_charts()*, among others.

6.6 0.2.0 (2019-12-11)

- It actually works!

6.7 0.1.0 (2019-12-10)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search