
Jupyter Commons Documentation

Release 0.2.1

Stefan Braun

Dec 22, 2020

Contents:

1	Jupyter Commons	1
1.1	Features	1
1.2	Credits	1
2	Installation	3
2.1	Stable release	3
2.2	From sources	3
3	Usage	5
4	Contributing	7
4.1	Types of Contributions	7
4.2	Get Started!	8
4.3	Pull Request Guidelines	9
4.4	Tips	9
4.5	Deploying	9
5	Indices and tables	11

Commonly used helpers for jupyter notebooks.

- Free software: BSD license
- Documentation: <https://jupyter-commons.readthedocs.io>.

1.1 Features

Functions usable in general Jupyter notebooks. Keep repeated implementation of helpers out of the notebook and re-use them easily.

1.2 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

ma.. highlight:: shell

2.1 Stable release

To install Jupyter Commons, run this command in your terminal:

```
$ pip install jupyter_commons
```

This is the preferred method to install Jupyter Commons, 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 Jupyter Commons can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/stbraun/jupyter_commons
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/stbraun/jupyter_commons/tarball/master
```

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

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


CHAPTER 3

Usage

To use Jupyter Commons in a project:

```
import jupyter_commons
```


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/stbraun/jupyter_commons/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

Jupyter Commons could always use more documentation, whether as part of the official Jupyter Commons 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/stbraun/jupyter_commons/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 *jupyter_commons* for local development.

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

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

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

```
$ cd jupyter_commons/  
$ pipenv --python 3.8  
$ pipenv install --dev
```

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 nox:

```
$ flake8 jupyter_commons tests  
$ python setup.py test or py.test  
$ nox
```

To get flake8 and nox, 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 versions as specified in setup.py. Check https://travis-ci.org/stbraun/jupyter_commons/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests.test_jupyter_commons
```

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:

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

Travis will then deploy to PyPI if tests pass.

CHAPTER 5

Indices and tables

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