

---

# JsonStreamReader Documentation

*Release 0.2.0*

**Matt Chapman**

**Jun 19, 2019**



---

## Contents:

---

<b>1</b>	<b>Json Stream Reader</b>	<b>1</b>
1.1	Features . . . . .	1
1.2	Credits . . . . .	1
<b>2</b>	<b>Installation</b>	<b>3</b>
2.1	Stable release . . . . .	3
2.2	From sources . . . . .	3
<b>3</b>	<b>Usage</b>	<b>5</b>
<b>4</b>	<b>Contributing</b>	<b>7</b>
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
<b>5</b>	<b>Credits</b>	<b>11</b>
5.1	Development Lead . . . . .	11
5.2	Contributors . . . . .	11
<b>6</b>	<b>History</b>	<b>13</b>
6.1	0.1.0 (2018-10-14) . . . . .	13
<b>7</b>	<b>Indices and tables</b>	<b>15</b>



# CHAPTER 1

---

## Json Stream Reader

---

Given a byte stream of {}-bracketed objects, generate one object-string at a time.

- Free software: GNU General Public License v3
- Documentation: <https://json-stream-reader.readthedocs.io>.

### 1.1 Features

- Turns a stream of unseperated {}-bracketed character (byte) strings, such as JSON serialized objects, into a collection of individual (byte)strings.

### 1.2 Credits

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



### 2.1 Stable release

To install JsonStreamReader, run this command in your terminal:

```
$ pip install json_stream_reader
```

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

You can either clone the public repository:

```
$ git clone git://github.com/matt2000/json_stream_reader
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/matt2000/json_stream_reader/tarball/master
```

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

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





## CHAPTER 3

---

### Usage

---

To use `JsonStreamReader` in a project:

```
import json_stream_reader
```



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/matt2000/json\\_stream\\_reader/issues](https://github.com/matt2000/json_stream_reader/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

JsonStreamReader could always use more documentation, whether as part of the official JsonStreamReader 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/matt2000/json\\_stream\\_reader/issues](https://github.com/matt2000/json_stream_reader/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 *json\_stream\_reader* for local development.

1. Fork the *json\_stream\_reader* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/json_stream_reader.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 json_stream_reader
$ cd json_stream_reader/
$ 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 json_stream_reader tests
$ python setup.py test or py.test
$ 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 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check [https://travis-ci.org/matt2000/json\\_stream\\_reader/pull\\_requests](https://travis-ci.org/matt2000/json_stream_reader/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_json_stream_reader
```

## 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.



### 5.1 Development Lead

- Matt Chapman <[matt@ninjitsuweb.com](mailto:matt@ninjitsuweb.com)>

### 5.2 Contributors

None yet. Why not be the first?





#### 6.1 0.1.0 (2018-10-14)

- First release on PyPI.



## CHAPTER 7

---

### Indices and tables

---

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