Swagger Aggregator Documentation

Release 0.1.1

Cyprien Guillemot

January 31, 2016

Contents:

CHAPTER 1

swagger-aggregator

Swagger-aggregator allow you to create a swagger REST API from several other swagger REST APIs.

This can be really useful if you want to make an API Gateway accessing some of your internal APIs. You can also filter which path you want to deliver, and which properties of your definitions you don't want to show.

1.1 Example Usage

Here is an example of an aggregate configuration.

```
args: pet_url

info:
  version: "0.1"
  title: "API Gateway"

basePath: /v2

apis:
    pet: http://pet_url/v2

exclude_paths:
  - DELETE /pets/{petId}

exclude_fields:
  petPet:
    - id
```

This is not the most useful aggregation, as it only aggregate one API. The first part, *args*, define that the first parameter we will send to the aggregate will be pet_url. Then pet_url will be replaced by the given value everywhere in the config. The two next part, *info* and *basePath*, are the same as the ones you can find in every swagger API. *apis*, define the different APIs you want to aggregate. A name is associated with it URL. Then *exclude_paths* allow you to not deliver some path. In this case we don't want the user to delete a pet.

Finally, *exclude_fields* define the attributes of the definitions we do not want to show. The value of the keys is the name of the API followed by the name of the definition. The value of each key will be a list of all properties to exclude.

Then use this command to generate the aggregated swagger file:

```
from traxit_aggregator import SwaggerAggregator

SwaggerAggregator('config.yaml', 'pet.com')
```

1.2 Documentation

More documentation is available at https://swagger-aggregator.readthedocs.org/en/latest/.

1.3 Setup

make install or *pip install swagger-aggregator*

1.4 License

swagger-aggregator is licensed under http://opensource.org/licenses/MIT.

CHAPTER 2

Installation

At the command line:

```
$ easy_install swagger_aggregator
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv swagger_aggregator
$ pip install swagger_aggregator
```

CHAPTER 3

Usage

To use Swagger Aggregator in a project:

Here is an example of an aggregate configuration.

```
args: pet_url

info:
  version: "0.1"
  title: "API Gateway"

basePath: /v2

apis:
    pet: http://pet_url/v2

exclude_paths:
  - DELETE /pets/{petId}

exclude_fields:
  petPet:
    - id
```

This is not the most useful aggregation, as it only aggregate one API. The first part, *args*, define that the first parameter we will send to the aggregate will be pet_url. Then pet_url will be replaced by the given value everywhere in the config. The two next part, *info* and *basePath*, are the same as the ones you can find in every swagger API. *apis*, define the different APIs you want to aggregate. A name is associated with it URL. Then *exclude_paths* allow you to not deliver some path. In this case we don't want the user to delete a pet.

Finally, *exclude_fields* define the attributes of the definitions we do not want to show. The value of the keys is the name of the API followed by the name of the definition. The value of each key will be a list of all properties to exclude.

Then use this command to generate the aggregated swagger file:

```
from traxit_aggregator import SwaggerAggregator

SwaggerAggregator('config.yaml', 'pet.com')
```

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/cyprieng/swagger_aggregator/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” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

4.1.4 Write Documentation

Swagger Aggregator could always use more documentation, whether as part of the official Swagger Aggregator 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/cyprieng/swagger_aggregator/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 *swagger_aggregator* for local development.

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

```
$ git clone git@github.com:your_name_here/swagger_aggregator.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 swagger_aggregator
$ cd swagger_aggregator/
$ 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 swagger_aggregator tests
$ python setup.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 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/cyprieng/swagger_aggregator/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_swagger_aggregator
```

CHAPTER 5

Credits

5.1 Development Lead

- Cyprien Guillemot <cyprien.guillemot@gmail.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.1.1 (2016-1-31)

- Change license to MIT.

6.2 0.1 (2016-1-29)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search