REST Header Pagination Documentation

Release 0.1.0

Sunscrapers

Feb 05, 2019

Contents:

Note: This is pre-alpha code. Use at your own discretion.

Hedju is a collection of classes that are designed to replace original DRF pagination classes and provide pagination using Link header. If client somehow does not support Link header, it can still ask for enveloped data.

Developed by SUNSCRAPERS with passion & patience.

CHAPTER 1

Hedju - Header Pagination for Django REST Framework

Replacement for Django REST Framework's pagination classes implementing Link header as defined in RFC5988 with optional enveloping.

Inspired by **'this paragraph from excellent article<https://www.vinaysahni.com/best-practices-for-a-pragmatic-restful-api#pagination>'_** by Vinay Sahni

Note: This is pre-alpha code. Use at your own discretion.

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

1.1 Features

- Provides `first`, `prev`, `next` and `last` links via headers.
- Optional enveloping for clients without header support - returns structure compatible with original class but with extra `first` and `last` links.

1.2 Credits

Developed by SUNSCRAPERS with passion & patience.

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

CHAPTER 2

Installation

2.1 Stable release

To install REST Header Pagination, run this command in your terminal:

```
$ pip install hedju
```

This is the preferred method to install REST Header Pagination, 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 REST Header Pagination can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/sunscrapers/hedju
```

Or download the tarball:

```
$ curl  -OL https://github.com/sunscrapers/hedju/tarball/master
```

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

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

CHAPTER 3

Usage

To use REST Header Pagination in a project, choose your preferred pagination style (see below). Each time you will have `envelope` param available. If you set it to true (or 1), you'll get original behavior of respective DRF's class but the Headers will also be set.

3.1 HeaderPageNumberPagination

Request:: GET https://api.example.com/accounts/?page=2

Response:

```
HTTP 200 OK

Headers:
    Link: <https://api.example.com/accounts/>; rel="first", <https://api.example.com/
↪accounts/?page=1>; rel="prev", <https://api.example.com/accounts/?page=3>; rel="next
↪", <https://api.example.com/accounts/?page=9>; rel="last"

Body:
[
    {
        "id": 1,
        "name": "John Doe",
    },
    ...
]
```

Request:: GET https://api.example.com/accounts/?page=2&envelope=true

Response:

```
HTTP 200 OK

Headers:
```

(continues on next page)

(continued from previous page)

```
    Link: <https://api.example.com/accounts/?envelope=true>; rel="first", <https://
↪api.example.com/accounts/?page=1>; rel="prev", <https://api.example.com/accounts/?
↪page=3>; rel="next", <https://api.example.com/accounts/?page=9>; rel="last"

Body:

{
    "count": 882,
    "first": "https://api.example.com/accounts/?envelope=true",
    "previous": "https://api.example.com/accounts/?page=1&envelope=true",
    "next": "https://api.example.com/accounts/?page=3&envelope=true",
    "last": "https://api.example.com/accounts/?page=9&envelope=true",
    "results": [
        {
            "id": 1,
            "name": "John Doe",
        },
        ...
    ]
}
```

3.1.1 Setup

To enable the LimitOffsetPagination style globally, use the following configuration:

```
REST_FRAMEWORK = {
    ...
    'DEFAULT_PAGINATION_CLASS': 'hedju.HeaderPageNumberPagination',
    'PAGE_SIZE': 100,
}
```

3.2 HeaderLimitOffsetPagination

Request:: GET https://api.example.com/accounts/?limit=100&offset400

Response:

```
HTTP 200 OK

Headers:
    Link: <https://api.example.com/accounts/?limit=100>; rel="first", <https://api.
↪example.com/accounts/?limit=100&offset=300>; rel="prev", <https://api.example.com/
↪accounts/?limit=100&offset=500>; rel="next", <https://api.example.com/accounts/?
↪limit=100&offset=782>; rel="last"

Body:
[
    {
        "id": 1,
        "name": "John Doe",
    },
    ...
]
```

Request:: GET https://api.example.com/accounts/?limit=100&offset400&envelope=true

Response:

```
HTTP 200 OK

Headers:
    Link: <https://api.example.com/accounts/?limit=100&envelope=true>; rel="first",
↪<https://api.example.com/accounts/?limit=100&offset=300&envelope=true>; rel=
↪"previous", <https://api.example.com/accounts/?limit=100&offset=500&envelope=true>;␣
↪rel="next", <https://api.example.com/accounts/?limit=100&offset=782&envelope=true>;␣
↪rel="last"

Body:

{
    "count": 882,
    "first": "https://api.example.com/accounts/?limit=100&envelope=true",
    "previous": "https://api.example.com/accounts/?limit=100&offset=300&envelope=true
↪",
    "next": "https://api.example.com/accounts/?limit=100&offset=500&envelope=true",
    "last": "https://api.example.com/accounts/?limit=100&offset=782&envelope=true",
    "results": [
        {
            "id": 1,
            "name": "John Doe",
        },
        ...
    ]
}
```

3.2.1 Setup

To enable the LimitOffsetPagination style globally, use the following configuration:

```
REST_FRAMEWORK = {
    ...
    'DEFAULT_PAGINATION_CLASS': 'hedju.HeaderLimitOffsetPagination',
    'PAGE_SIZE': 100,  # Optional
}
```

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/sunscrapers/hedju/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

REST Header Pagination could always use more documentation, whether as part of the official REST Header Pagination 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/sunscrapers/hedju/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 *hedju* for local development.

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

```
$ git clone git@github.com:your_name_here/hedju.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 hedju
$ cd hedju/
$ 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 hedju 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/sunscrapers/hedju/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_hedju
```

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

Credits

5.1 Development Lead

- Dominik Kozaczko <d.kozaczko@sunscrapers.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.1.0 (2019-02-05)

- Add remaining pagination classes

6.2 0.0.4 (2019-02-05)

- Working tests
- Some doc updates

6.3 0.0.1 (2019-02-04)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search