marshmallow-union

Release 0.1.14

Feb 09, 2020

Contents

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 1

Overview

docs	
tests	
package	

Union fields for marshmallow.

- Free software: MIT license

1.1 Warning

> Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

1.2 Documentation

https://python-marshmallow-union.readthedocs.io/

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 2

Installation

At the command line:

```
pip install marshmallow-union
```

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 3

Usage

To use marshmallow-union in a project import marshmallow_union.Union and pass it a list of marshmallow.fields.Field instances.

When deserializing, each field will be tried in sequence until one succeeds, not raising marshmallow.exceptions.ValidationError. If none of them is successful, marshmallow.exceptions.ValidationError is raised.

When serializing, each field will be tried in sequence until one succeeds, not raising *any* exception. If none of them is successful, *marshmallow_union.ExceptionGroup* is raised, containing the values of the raised exceptions.

See the API reference for more details.

```
import marshmallow
import marshmallow_union


class PersonSchema(marshmallow.Schema):
    name = marshmallow.fields.String()
    number_or_numbers = marshmallow_union.Union(
        [
            marshmallow.fields.List(marshmallow.fields.Integer()),
            marshmallow.fields.Integer(),
        ],
            reverse_serialize_candidates=True,
    )



input_data = {"name": "Alice", "number_or_numbers": 25}
schema = PersonSchema()

loaded = schema.load(input_data)
dumped = schema.dump(loaded)

assert dumped == input_data
```

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 4

Reference

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

4.1 marshmallow_union package

4.1.1 Module contents

Union fields for marshmallow.

exception `marshmallow_union.`**ExceptionGroup**(*msg*, *errors*)

Bases: *marshmallow_union.MarshmallowUnionException*

Collection of possibly multiple exceptions.

exception `marshmallow_union.`**MarshmallowUnionException**

Bases: `Exception`

Base exception for marshmallow_union.

class `marshmallow_union.`**Union**(*fields*, *reverse_serialize_candidates=False*, ***kwargs*)

Bases: `marshmallow.fields.Field`

Field that accepts any one of multiple fields.

Each argument will be tried until one succeeds.

Parameters

- **fields** (`List[Field]`) – The list of candidate fields to try.
- **reverse_serialize_candidates** (`bool`) – Whether to try the candidates in reverse order when serializing.

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 5

Contributing

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

5.1 Bug reports

When 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.

5.2 Documentation improvements

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

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/adamboche/python-marshmallow-union/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 code contributions are welcome :)

5.4 Development

To set up *python-marshmallow-union* for local development:

1. Fork python-marshmallow-union (look for the "Fork" button).
2. Clone your fork locally:

   ```
   git clone git@github.com:your_name_here/python-marshmallow-union.git
   ```

3. Create a branch for local development:

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

   Now you can make your changes locally.
4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

   ```
   tox
   ```

5. 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
   ```

6. Submit a pull request through the GitHub website.

5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a file in `changelog.d/` describing the changes. The filename should be `{id}.{type}.rst`, where `{id}` is the number of the GitHub issue or pull request and `{type}` is one of `breaking` (for breaking changes), `deprecation` (for deprecations), or `change` (for non-breaking changes). For example, to add a new feature requested in GitHub issue #1234, add a file called `changelog.d/1234.change.rst` describing the change.
4. Add yourself to `AUTHORS.rst`.

5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.

It will be slower though . . .

```
detox
```

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 6

Authors

- Adam Boche - https://github.com/adamboche

Warning: Figuring out the correct field type from an untagged dump is a matter of guesswork and can be error-prone. A more explicit approach like https://github.com/Bachmann1234/marshmallow-polyfield/ is better. This project remains here to avoid breaking existing software.

CHAPTER 7

0.1.12 (2019-10-24)

7.1 Backward-incompatible Changes

- On serialization, `marshmallow_union.ExceptionGroup` is raised if all candidate fields fail. #24

CHAPTER 8

0.1.11 (2019-06-19)

8.1 Changes

- Override the underscore-prefixed methods instead of the plain ones #22

8.1.1 Changelog

CHAPTER 9

0.1.10 (2019-06-08)

9.1 Changes

- Use Towncrier for changelog. #18

CHAPTER 10

0.1.0 (2019-06-07)

- First release on PyPI.

CHAPTER 11

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index