
marshmallow-union

Release 0.1.15

Adam Boche

Jan 07, 2022

CONTENTS

1 Overview	3
2 Installation	5
3 Usage	7
4 Reference	9
5 Contributing	11
6 Authors	15
7 0.1.15 (2020-05-30)	17
8 0.1.12 (2019-10-24)	19
9 0.1.11 (2019-06-19)	21
10 0.1.10 (2019-06-08)	23
11 0.1.0 (2019-06-07)	25
12 Indices and tables	27
Python Module Index	29
Index	31

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.

OVERVIEW

docs	
tests	
package	

Union fields for marshmallow.

- Free software: MIT license

1.1 Warning

This library works by trying a list of fields one by one, and (de)serializing with the first one not to raise an error. The type of the values is not taken into account, so if one of the fields in the union accepts values of an unexpected type, they will be used for serialization. This can lead to a surprising behavior, because :

```
u = Union(fields=[fields.Integer(), fields.String()]) # the Integer field
↳ accepts string representations of integers
type(u.deserialize('0')) # -> int
```

If you want to have precise control of which field will be used for which value, you can use [marshmallow-polyfield](#) instead of this library.

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.

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.

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.

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.

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*.

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 ...

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`):

```
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.

AUTHORS

- Adam Boche - <https://github.com/adamboche>
- Ophir Lojkine - <https://github.com/lovasoa>

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.

0.1.15 (2020-05-30)

7.1 Changes

- Fix #32 : Never ignore the value passed to `_serialize`. Notably, this fixes the serialization of lists of unions, such as `List(Union([Int(), String()]))`

0.1.12 (2019-10-24)

8.1 Backward-incompatible Changes

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

0.1.11 (2019-06-19)

9.1 Changes

- Override the underscore-prefixed methods instead of the plain ones [#22](#)
-

9.1.1 Changelog

0.1.10 (2019-06-08)

10.1 Changes

- Use Towncrier for changelog. #18
-

0.1.0 (2019-06-07)

- First release on PyPI.

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

m

marshmallow_union, 9

INDEX

E

ExceptionGroup, 9

M

marshmallow_union

 module, 9

MarshmallowUnionException, 9

module

 marshmallow_union, 9

U

Union (*class in marshmallow_union*), 9