
Value Object Documentation

Release 0.3.1

Paweł Zadrożny

Mar 22, 2018

Contents:

1	Value Object	3
1.1	Features	3
1.2	Installation	3
1.3	Documentation	3
1.4	Quick Example	4
2	Public API	5
2.1	Instance of Value	5
2.2	Modification forbidden error	7
3	Credits	9
3.1	Development	9
3.2	Contributors	9
4	Contributing	11
4.1	Types of Contributions	11
4.2	Get Started!	12
4.3	Pull Request Guidelines	13
5	LICENSE	15
	Python Module Index	17

Value Object is an attempt to make DDD implementation of Value. It's not the same as Data Class proposed in **PEP 557** <https://www.python.org/dev/peps/pep-0557/> but shares some similarities like frozen attributes.

Info DDD Value Object implementation.

Author Paweł Zadrozny @pawelzny <pawel.zny@gmail.com>

1.1 Features

- Value object can't be changed once created
- Two objects with the same values are considered equal
- Access to values after dot: `value.my_value`
- Access to values like dict: `value['my_value']`

1.2 Installation

```
pip install vo
```

Package: <https://pypi.org/project/vo/>

1.3 Documentation

Read full documentation on: <http://vo.readthedocs.io>

1.4 Quick Example

```
>>> from vo import Value
...
>>> class Book(Value):
...     price = None
...     title = None
...     publisher = None
...
>>> book = Book(price=120, title='Python for masses', publisher='SPAM')
>>> book
Book(price=120, title='Python for masses', publisher='SPAM')

>>> book.title
'Python for masses'

>>> book['price']
120

>>> book.price = 230
Traceback (most recent call last):
  File "<input>", line 1, in <module>
    raise ImmutableInstanceError
vo.value.ImmutableInstanceError: Modification of Value instance is forbidden.
```


2.1 Instance of Value

class vo.value.Value (**kwargs)

Basic implementation of DDD immutable Value Object.

Features:

- Two objects with exact values are considered the same
- Objects are immutable (raise ImmutableInstanceError)

Example

```
>>> val = Value(name='Primary', price=10.35, currency='USD')
>>> val.price
10.35

>>> val['currency']
'USD'
```

Parameters ****kwargs** – Any key=value pairs.

to_dict () → dict

Dump values to dict.

Returns dict with values

Return type dict

to_json () → str

Dump values to JSON.

Returns JSON string.

Return type str

`to_bytes()` → bytes
Convert values to bytes.

Returns byte string

Return type bytes

2.1.1 Create new instance of Value

Value accept any key=value pairs. These pairs will be attached to object as attributes. Once created values are immutable. Can't be changed or deleted.

```
>>> from vo import Value
>>> book = Value(title='Learning Python',
...             authors=['Mark Lutz', 'David Ascher'],
...             publisher="O'REILLY")
>>> book
<vo.value.Value object at 0x7f38862b3860>
```

2.1.2 Values access

Values can be accessed like object attributes or like dict keys.

```
>>> from vo import Value
>>> book = Value(title='Learning Python',
...             authors=['Mark Lutz', 'David Ascher'],
...             publisher="O'REILLY")
>>> book.title == book['title']
True

>>> book.authors == book['authors']
True
```

2.1.3 Objects comparison

Let's take the same book example.

```
>>> from vo import Value
>>> book1 = Value(title='Learning Python',
...              authors=['Mark Lutz', 'David Ascher'],
...              publisher="O'REILLY")
>>> book2 = Value(title='Learning Python',
...              authors=['Mark Lutz', 'David Ascher'],
...              publisher="O'REILLY")
>>> book1 == book2
True

>>> book1 is book2
False
```

2.1.4 Value lookup

Check if value exists.

```

>>> from vo import Value
>>> book = Value(title='Learning Python',
...             authors=['Mark Lutz', 'David Ascher'],
...             publisher="O'REILLY")
>>> 'title' in book
True

>>> 'price' in book
False

>>> book.title
'Learning Python'

>>> book.price
Traceback (most recent call last):
  File "<input>", line 1, in <module>
AttributeError: 'Value' object has no attribute 'price'

```

2.2 Modification forbidden error

exception `vo.value.ImmutableInstanceError` (*message: str = None*)

Raised on any attempt to modify values in Value object.

Parameters `message` (*str*) – Optional message.

`message = 'Modification of Value instance is forbidden.'`

Any attempt of value modification or delete will raise *ImmutableInstanceError*

```

>>> from vo import Value
>>> book = Value(title='Learning Python',
...             authors=['Mark Lutz', 'David Ascher'],
...             publisher="O'REILLY")
>>> book.title = 'Spam'
Traceback (most recent call last):
  File "<input>", line 1, in <module>
    raise ImmutableInstanceError()
vo.value.ImmutableInstanceError: Modification of Value frozen instance is forbidden.

```


3.1 Development

- Paweł Zadrozny @pawelzny <pawel.zny@gmail.com>

3.2 Contributors

None yet. Why not be the first?

Read more how to contribute on *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/pawelzny/vo/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

authentication could always use more documentation, whether as part of the official authentication 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/pawelzny/vo/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 *vo* for local development.

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

```
$ git clone git@github.com:your_name_here/vo.git
```

3. Install your local copy into a virtualenv. Assuming you have PipEnv installed, this is how you set up your fork for local development:

```
$ cd vo/  
$ make install-dev
```

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:

```
$ make test-all
```

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 3.4, 3.5, 3.6, and for PyPy3.5. Check <https://circleci.com/gh/pawelzny/vo> and make sure that the tests pass for all supported Python versions.

CHAPTER 5

LICENSE

ISC License

Copyright (c) 2017, Paweł Zadrożny @pawelzny <pawel.zny@gmail.com>

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED “AS IS” AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

V

`vo.value`, 7

I

`ImmutableInstanceError`, 7

M

`message` (`vo.value.ImmutableInstanceError` attribute), 7

T

`to_bytes()` (`vo.value.Value` method), 5

`to_dict()` (`vo.value.Value` method), 5

`to_json()` (`vo.value.Value` method), 5

V

`Value` (class in `vo.value`), 5

`vo.value` (module), 5, 7