Value Object Documentation

Release 0.3.0

Paweł Zadrożny

Mar 17, 2018

Contents:

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.

CHAPTER 1

Value Object

Info DDD Value Object implementation.

Author Paweł Zadrożny @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
>>> value = Value(test=True, some_text="I am some text string")
>>> value == value
True

>>> value_clone = Value(some_text="I am some text string", test=True)
>>> value == value_clone
True

>>> value is value_clone
False

>>> value_truth = Value(purpose_of_life=42)
>>> value == value_truth
False

>>> value_truth.purpose_of_life
42
>>> value_truth['purpose_of_life']
42
```

CHAPTER 2

Public API

2.1 Instance of Value

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

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.
```

CHAPTER 3

Credits

3.1 Development

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

3.2 Contributors

None yet. Why not be the first?

Read more how to contribute on *Contributing*.

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

CHAPTER 5

LICENSE

Python Module Index

Index