Autodoc Documentation

Release 0.1

Shinya Ohyanagi

Nov 02, 2017

Contents

CHAPTER 1

What is Autodoc

Generate documentation from your unit-test.

This library is Python implementation of Autodoc.

- Autodoc
- Test::JsonAPI::Autodoc

CHAPTER 2

Installation

```
$ virtualenv --distribute autodoc_sample
$ source autodoc/bin/activate
$ cd autodoc
$ pip install autodoc
```

CHAPTER 3

Usage

Run unittest with PYAUTODOC=1 to generate documents for your tests decorated with *@autodoc.generate*.

```
PYAUTODOC=1 python -m unittest examples/test_unittest.py
```

If you use py.test as test runner.

```
PYAUTODOC=1 py.test tests examples/test_pytest.py
```

If you use nose as test runner.

```
PYAUTODOC=1 nosetests tests examples/test_unittest.py
```

CHAPTER 4

Example for unittest

```
class TestUnittest(TestCase):
  def setUp(self):
    app = create_app
    self.client = TestApp(app)

  @classmethod
  @autodoc.generate('var/test_unittest.rst')
  def tearDownClass(cls):
    pass

  @autodoc.describe('GET /')
  def test_get(self):
    """ GET / """
    res = self.client.get('/')
    self.assertEqual(res.status_code, 200)

    return res
```

@autodoc.describe() describe test name.

For example *GET /* assigned to generated document.

@autodoc.generate(path_to_output) will generate document.

CHAPTER 5

Example for py.test

```python
@pytest.fixture
def setup():
  setup = TestApp(create_app)

  return setup


@autodoc.generate('var/test_pytest.md', template='templates/markdown.md')
def teardown_module(module):
  pass


@autodoc.describe('POST /')
def test_post(setup):
  res = setup.post_json('/', params={'id': 1, 'message': 'foo'})
  assert res.status_code == 200

  return res
```

CHAPTER 6

Example for requests

CHAPTER 7

Conventions

7.1 Return WebTest or requests response in test method

Py-Autodoc must return WebTest response or requests response.

```
@autodoc.describe('POST /')
def test_post(setup):
  res = setup.post_json('/', params={'id': 1, 'message': 'foo'})
  assert res.status_code == 200

  return res # Must return WebTest or requests response.
```

7.2 Generate document point

@autodoc.generate will create document.

If you set *@autodoc.generate* to each test case, document will generate each file.

```
class TestUnittest(TestCase):
  def setUp(self):
    app = create_app
    self.client = TestApp(app)

  @autodoc.generate('var/indext_get.rst')
  @autodoc.describe('GET /')
  def test_get(self):
    """ GET / """
    res = self.client.get('/')
    self.assertEqual(res.status_code, 200)

    return res
```

```
@autodoc.generate('var/foo_get.rst')
@autodoc.describe('GET /foo')
def test_get(self):
  """ GET / """
  res = self.client.get('/foo')
  self.assertEqual(res.status_code, 200)

  return res
```

This will generate *var/index_get.rst* and *var/foo_get.rst*.

If you want to generate all tests into single file, decorate *@autodoc.generate* to *tearDownClass*, *teardown_module* fixture.

CHAPTER 8

Configuration

You can configure @*autodoc.generat(output, template=path_to_template)* to change template file.

CHAPTER 9

Indices and tables

- genindex
- modindex
- search