

---

# **rf-django-misc**

***Release 0.1.1***

**Nov 10, 2017**



---

## Contents

---

<b>1</b>	<b>Overview</b>	<b>1</b>
1.1	Installation . . . . .	1
1.2	Documentation . . . . .	1
1.3	Development . . . . .	1
<b>2</b>	<b>Installation</b>	<b>3</b>
<b>3</b>	<b>Usage</b>	<b>5</b>
<b>4</b>	<b>Reference</b>	<b>7</b>
4.1	rf_django_misc . . . . .	7
<b>5</b>	<b>Contributing</b>	<b>9</b>
5.1	Bug reports . . . . .	9
5.2	Documentation improvements . . . . .	9
5.3	Feature requests and feedback . . . . .	9
5.4	Development . . . . .	10
<b>6</b>	<b>Authors</b>	<b>11</b>
<b>7</b>	<b>Changelog</b>	<b>13</b>
7.1	0.1.0 (2016-12-12) . . . . .	13
<b>8</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>



# CHAPTER 1

---

## Overview

---

docs	
tests	
package	

This package contains various useful bits of code for use in Django projects. I consider the things here to be too small to warrant a package of their own, but I don't want to copy and paste them from project to project. I will accept bug reports and pull requests for this package, but I make no promise to maintain it or keep any kind of backwards compatibility. Use at your own risk.

- Free software: BSD license

## 1.1 Installation

```
pip install rf-django-misc
```

## 1.2 Documentation

<https://rf-django-misc.readthedocs.io/>

## 1.3 Development

To run the all tests run:

```
tox
```

Note, to combine the coverage data from all the tox environments run:

Windows	<pre>set PYTEST_ADDOPTS=--cov-append tox</pre>
Other	<pre>PYTEST_ADDOPTS=--cov-append tox</pre>

## CHAPTER 2

---

### Installation

---

At the command line:

```
pip install rf-django-misc
```





## CHAPTER 3

---

### Usage

---

To use rf-django-misc in a project:

```
import rf_django_misc
```



## 4.1 rf\_django\_misc

`rf_django_misc.test.get_test_request` (*url*='/', *data*=None, *method*='get', *session\_data*=None, *user*=None)

Get a test request for testing a Django view

This is helpful if you want to test a view as an isolated unit, i.e. not with the usual Django test client. This function returns a test request generated by `RequestFactory`, with the `session` and `user` attributes set.

### Parameters

- **url** – The URL for the request
- **data** – Request data
- **method** – Request method
- **session\_data** – Session data
- **user** – A user object



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

rf-django-misc could always use more documentation, whether as part of the official rf-django-misc 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/rfleschenberg/rf-django-misc/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 *rf-django-misc* for local development:

1. Fork [rf-django-misc](#) (look for the “Fork” button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/rf-django-misc.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`)<sup>1</sup>.
2. Update documentation when there’s new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

### 5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

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

```
detox
```

---

<sup>1</sup> 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 ...

## CHAPTER 6

---

### Authors

---

- René Fleschenberg - <https://fleschenberg.net>





## CHAPTER 7

---

### Changelog

---

#### 7.1 0.1.0 (2016-12-12)

- First release on PyPI.



## CHAPTER 8

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### **r**

`rf_django_misc`, [7](#)  
`rf_django_misc.drf`, [7](#)  
`rf_django_misc.test`, [7](#)



## G

`get_test_request()` (in module `rf_django_misc.test`), [7](#)

## R

`rf_django_misc` (module), [7](#)

`rf_django_misc.drf` (module), [7](#)

`rf_django_misc.test` (module), [7](#)