django-user-messages Documentation

Release 0.7-1-gdd76bb0

Feinheit AG

Aug 17, 2021

Contents

1 Installation **3**

2 Usage **5**

3 Change log **7**

3.1 Next version . . . 7
3.2 0.7 (2020-01-22) . . . 7
3.3 0.6 (2018-09-26) . . . 7
3.4 0.5 (2018-03-04) . . . 7
3.5 0.4 (2017-07-19) . . . 8
3.6 0.3 (2017-05-18) . . . 8
3.7 0.2 (2017-05-18) . . . 8
3.8 0.1 (2017-05-17) . . . 8

django-user-messages adds offline messaging support to Django's messaging framework. It achieves this by allowing to save messages in the database. The `user_messages.api.get_messages` utility and the `user_messages.context_processors.messages` context processor transparently concatenate Django's messages and our own messages in a single list, therefore existing code works without any changes and without causing database writes. django-user-messages' functions have to be used explicitly. I consider this a feature, not a bug.

CHAPTER 1

Installation

- Install `django-user-messages` using pip into your virtualenv.
- Add `user_messages` to `INSTALLED_APPS` and run `migrate`.
- Replace the default messages context processor with `user_messages.context_processors.messages`. The context processor provides both django.contrib.messages' and django-user-messages' messages. Note that Django 2.2's admin app checks for the presence of the default messages context processor, so you'll have to silence this check by adding `"admin.E404"` to the `SILENCED_SYSTEM_CHECKS` setting.
- Use `user_messages.api` as you would use `django.contrib.messages` except that you pass the user model or ID as first parameter, not the current request.

CHAPTER 2

Usage

Pretty much the same as Django's messaging framework:

```
from user_messages import api

api.info(user, 'Hey there')
api.warning(user, 'Stop this')
api.error(user, 'Not nice!')

# Passing the ID is also possible; the user instance does not
# have to be instantiated at all:
api.success(user.id, 'Yay!')
```

django-user-messages' messages supports two additional features not available in Django's messages framework:

- Messages can be delivered more than once by passing `deliver_once=False`. These messages have to be acknowledged explicitly. django-user-messages does not contain any code to do this.
- It is possible to attach additional data by passing a dictionary as `meta`:

  ```
  api.debug(user, 'Oww', meta={
      'url': 'http://example.com',
  })
  ```

For convenience, our messages have the same `tags` and `level_tag` properties as Django's messages. Meta properties are also accessible in templates:

```
{% if messages %}
  <ul class="messages">
  {% for message in messages %}
    <li class="{{ message.tags }}".>
      {% if message.meta.url %}<a href="{{ message.meta.url }}">{% endif %}
      {{ message }}
      {% if message.meta.url %}</a>{% endif %}
    </li>
  {% endfor %}
```

(continues on next page)

(continued from previous page)

```
  </ul>
{% endif %}
```

django-user-messages' messages are also evaluated lazily.

CHAPTER 3

Change log

3.1 Next version

- Switched to a declarative setup.
- Switched from Travis CI to GitHub actions.

3.2 0.7 (2020-01-22)

- Fixed a crash because of unicode strings being returned from `Message.__str__` in Python 2.
- Order messages upon retrieval.
- Added Django 3.0 to the Travis CI matrix.
- Replaced `ugettext*` with `gettext*` to avoid deprecation warnings.

3.3 0.6 (2018-09-26)

- Reformatted the code using black.
- Added a hint about silencing the messages context processor system check under Django 2.2.

3.4 0.5 (2018-03-04)

- Added german translations and a nice app name.
- Changed the implementation of keyword-only arguments to be compatible with Python 2.

3.5 0.4 (2017-07-19)

- **Backwards incompatible** Rebuilt the model to not use Django's `JSONField` at all. This design decision unnecessarily restricted the areas where django-user-messages was usable.
- Fixed properties to be more forgiving with missing data.
- Added tox configuration for running tests and coding style checks and for building the docs.
- Improved documentation and test coverage.

3.6 0.3 (2017-05-18)

- Added usage instructions.
- Merge the `message` and `meta` JSON fields into a single `data` field and imitate the `Message` object interface more closely.

3.7 0.2 (2017-05-18)

- Added the possibility to associate additional data with a message py passing a dictionary as the `meta` keyword-only argument to the API.
- Changed the module to import the `Message` model as late as possible so that the API can easily be imported for example in a `AppConfig` module.

3.8 0.1 (2017-05-17)

- Initial public release.