
django-template-helpers

unknown

May 13, 2021

CONTENTS

1	Features	3
2	Requirements	5
3	Prepare for development	7
3.1	Installation	7
3.2	Usage	7
3.3	Changelog	10
3.4	template_helpers.templatetags	10
3.5	template_helpers.context_processors	11
4	Indices and tables	13
	Python Module Index	15
	Index	17

django-template-helpers provides template tags to add missing features to the Django template language.

FEATURES

- Template tag to set a new context variable `set`
- Template filter to split a string into a list `split`
- Context processor to expose selected settings to the templates
- `GenericTemplateView` for static pages

REQUIREMENTS

django-template-helpers supports Python 3 only and requires at least Django 1.11.

PREPARE FOR DEVELOPMENT

A Python 3 interpreter is required in addition to poetry.

```
$ poetry install
```

Now you're ready to run the tests:

```
$ make tests
```

Contents:

3.1 Installation

- Install with pip:

```
pip install django-template-helpers
```

- Your `INSTALLED_APPS` setting:

```
INSTALLED_APPS = (  
    # ...  
    'template_helpers',  
)
```

3.2 Usage

3.2.1 Expose settings to template context

In order to expose settings to the template context, you have to add the `template_helpers.context_processors.settings` context processor to your `TEMPLATES` configuration block.

When done, add a list of settings to expose to your configuration.

```
TEMPLATE_EXPOSED_SETTINGS = (  
    'DEBUG',  
    'SOME_PUBLIC_API_TOKEN',  
)
```

3.2.2 Settings new template context variables

If you need to add new context variables within your templates, use the `set` tag.

```
{% load template_helpers %}

{% set foo="Ipsum" %}

Lorem {{ foo }}
```

The output will be

```
Lorem Ipsum
```

3.2.3 Splitting a string

To split a string for iteration, you can use the `split` filter. The filter allows to pass an alternative delimiter, default is ” “.

```
{% load template_helpers %}

{% for item in "item1,item2"|split:", " %}
    Item: {{ item }}
{% endfor %}
```

The output will be

```
Item: item1
Item: item2
```

3.2.4 Add extra span element inside text

To add extra `` element inside a string use `starspan` filter.

```
class SomeModel(models.Model):
    headline = models.CharField(help_text=_('Use ***my text*** to highlight "my text".'))
    ...
```

```
{% load template_helpers %}

{{ headline|starspan }}
```

3.2.5 Append a list to another list in template

If you need to join lists within your templates, use the `merge_list` filter.

```
{% load template_helpers %}

{% for element in first_list|merge_list:second_list %}
  {{ element }}
{% endfor %}
```

To make the result list persistent use `merge_list` filter in combination with `set` template tag.

```
{% load template_helpers %}

{% set new_list=first_list|merge_list:second_list %}
```

3.2.6 Add object attributes to context of included template

If you have an object with many attributes which need to be directly accessible in included template, use the `include_with` tag.

Suppose you have a `Person` model:

```
class Person(models.Model):
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

Define exposed attributes on it:

```
class Person(models.Model):
    template_exposed_attributes = ['first_name', 'last_name']
    ...
```

Then you can use `include_with` template tag:

```
{% load template_helpers %}

{% include_with person 'some/template.html' %}
```

Instead of

```
{% include 'some/template.html' with first_name=person.first_name last_name=person.last_
↪name %}
```

It is also possible to overwrite / add additional kwargs.

```
{% load template_helpers %}

{% include_with person 'some/template.html' first_name='Laurel' best_friend='Hardy' %}
```

3.2.7 Using GenericTemplateView

GenericTemplateView is a TemplateView extension, that allows including static pages. The template path is encoded in url as `template` keyword argument, and the templates base directory can be set with `template_base_dir` keyword argument in `GenericTemplateView.as_view` call.

The GenericTemplateView can be used e.g. for template testing.

```
if settings.DEBUG:
    urlpatterns += [
        url(
            r'^tests/(?P<template>[\w\-\./]+)?$',
            GenericTemplateView.as_view(template_base_dir='mytests')
        ),
```

If test templates are located in `templates/mytests/...` (e.g. `templates/mytests/base/buttons/buttons.html`) we can now hit them by calling e.g. `localhost:8000/tests/base/buttons/buttons` url.

If no `template_base_dir` or `template` are specified, the view will try to render `index.html`. For more elaborate behavior overwrite the `get_template_base_dir` and `get_template_names` methods.

3.3 Changelog

3.3.1 0.2.0 (2019-09-12)

- Add `include_with` templatetag
- Add `merge_list` filter
- Add `starspan` filter

3.3.2 0.1.0 (2019-01-16)

- Add GenericTemplateView

3.3.3 0.0.1 (2018-08-08)

- Initial release of *django-template-helpers*

Api documentation:

3.4 `template_helpers.templatetags`

`template_helpers.templatetags.template_helpers.set_tag(context, **kwargs)`

The `set`` template tag add one or more context variables to the current template context. The context stack is respected.

```
{% set foo="bar" baz=1 %}
{{ foo }}
{{ baz }}
```

`template_helpers.templatetags.template_helpers.split(value, sep='')`

The `split` template filter splits a given string by spaces. If you want to split by something else, provide the divider as a argument to the filter.

```
{{ "foo bar"|split }}
{{ "foo-bar"|split "-" }}
```

`template_helpers.templatetags.template_helpers.starspan(value)`

The `starspan` template filter adds extra span element to star indicated text (`""text*`).

```
{{ some_text_variable|starspan }}
```

`template_helpers.templatetags.template_helpers.merge_list(value, list_to_merge)`

The `merge_list` filter combines two lists.

```
{% for element in first_list|merge_list:second_list %}
  {{ element }}
{% endfor %}
```

To make the result list persistent use in combination with `set` tag.

```
{% set new_list=first_list|merge_lists:second_list %}
```

```
class template_helpers.templatetags.template_helpers.IncludeWithNode(with_object,
                                                                    template_name, *args,
                                                                    extra_context=None,
                                                                    **kwargs)
```

Bases: `django.template.loader_tags.IncludeNode`

render(*context*)

Render the specified template and context. Cache the template object in `render_context` to avoid reparsing and loading when used in a for loop.

`template_helpers.templatetags.template_helpers.do_include_with(parser, token)`

Include template with object attributes injected to context. It takes object and template path as arguments. Object should have `template_exposed_attributes` list defined.

```
{% include_with obj 'path/to/included/template.html' %}
```

It is also possible to overwrite / add additional kwargs.

```
{% include_with obj 'path/to/included/template.html' foo='bar' %}
```

3.5 template_helpers.context_processors

`template_helpers.context_processors.settings(request)`

The `settings` context processor adds all settings from the `TEMPLATE_EXPOSED_SETTINGS` configuration to the template context.

INDICES AND TABLES

- [genindex](#)
- [modindex](#)
- [search](#)

PYTHON MODULE INDEX

t

`template_helpers.context_processors`, 11

`template_helpers.templatetags.template_helpers`,
10

D

`do_include_with()` (in module `template_helpers.templatetags.template_helpers`),
11

I

`IncludeWithNode` (class in `template_helpers.templatetags.template_helpers`),
11

M

`merge_list()` (in module `template_helpers.templatetags.template_helpers`),
11

module

`template_helpers.context_processors`, 11
`template_helpers.templatetags.template_helpers`,
10

R

`render()` (`template_helpers.templatetags.template_helpers.IncludeWithNode`
method), 11

S

`set_tag()` (in module `template_helpers.templatetags.template_helpers`),
10

`settings()` (in module `template_helpers.context_processors`), 11

`split()` (in module `template_helpers.templatetags.template_helpers`),
10

`starspan()` (in module `template_helpers.templatetags.template_helpers`),
11

T

`template_helpers.context_processors`
module, 11

`template_helpers.templatetags.template_helpers`
module, 10