31 December 2019

# Custom Tag and Filters

You can extend the template engine by defining custom tags and filters using Python, and then make them available to your templates using the **{% load %}** tag

Directory Layout

```
polls/
    __init__.py
    models.py
    templatetags/
        __init__.py
        poll_extras.py        ( Can be anything )
    views.py
```

Then we can access our custom tags and filters in the templates using
{% load poll_extras %}

To be a valid tag library  it must have a module level variable called register which must be an instance of template.Library

```
From django import template
Register = template.Library()
```

Custom filters can take in two arguments

1. Value of the variable
2. The value of the argument (can be left out altogether)

For example, in the filter **{{ var|foo:"bar" }}**, the filter **foo** would be passed the variable **var** and the argument **“bar”**.

```
def cut(value, arg):
    """Removes all values of arg from the given string"""
    return value.replace(arg, '')
```

Once we have written our custom filter we need to register the filter in our library.

```
register.filter('cut', cut)
```

1st argument : Name of the filter
2nd argument : Python function to be called

We can also use decorator
@register to do this

@register(name="cut")

If the 1st argument is expected to be a string we can use

```
from django.template.defaultfilters import stringfilter

@stringfilter
```

To escape characters we can use the is_safe = True param to the register decorator