Django Templates Advanced

SoftUni Team

Technical Trainers

Software University

https://softuni.bg

Have a Question?

Software University

sli.do

#python-web

Table of Contents

Template Inheritance

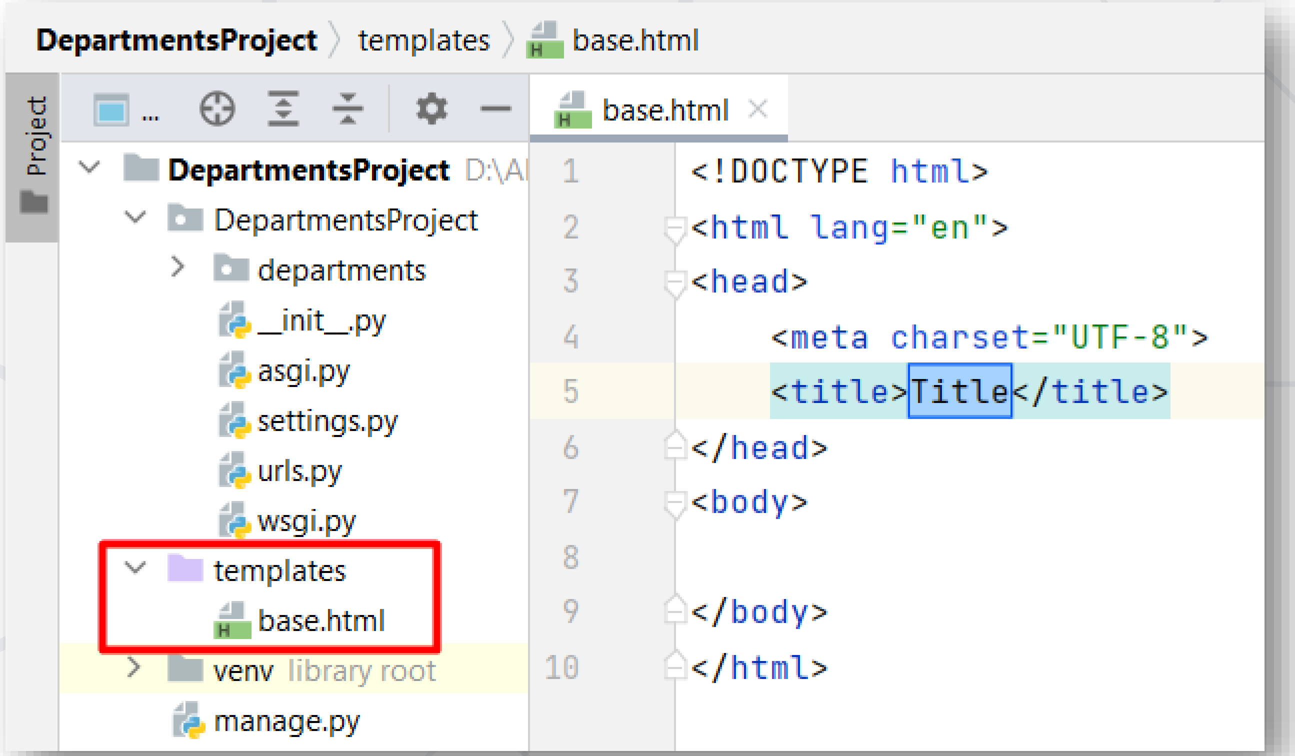
block, endblock, extends

Template Inheritance

- Template inheritance enables the creation of a **foundational** **skeleton** **template**
- This **base** **template** encompasses **shared** elements and establishes **blocks** that **child** templates can **customize**
- Generally, elements like the **header** and **footer** remain consistent throughout the entire application
- By leveraging **template** **inheritance**, we can **efficiently** **reuse** **common** components across different sections of our application

Example: Template Inheritance

- Create the **base** template in the **project/templates** directory
  - The base template is typically called `base.html`



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>

</body>
</html>
```

Example: Template Inheritance

base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
</head>

<body>
    {% block content %}
        # default code if no content is passed in
    {% endblock %}
</body>
</html>
```

You can have many blocks in one template

Each block should have its own unique name

Example: Template Inheritance

index.html

```
{% extends "base.html" %}

{% block title %}
    Home Page
{% endblock %}

{% block content %}
    {% for dep in departments.items %}
        <h2>{{ dep.title }}</h2>
        <p>{{ dep.description }}</p>
    {% endfor %}
{% endblock %}
```

Use the extends tag in the "child" templates

"Home Page" will be injected into the title block in the base.html

The code will be injected into the content block in the base.html

Including Template Snippets

- Template **snippets** provide a way to **include** templates within another template

```
{% include template-name %}
```

- The **template name** for **inclusion** can be specified using
  - a **variable**

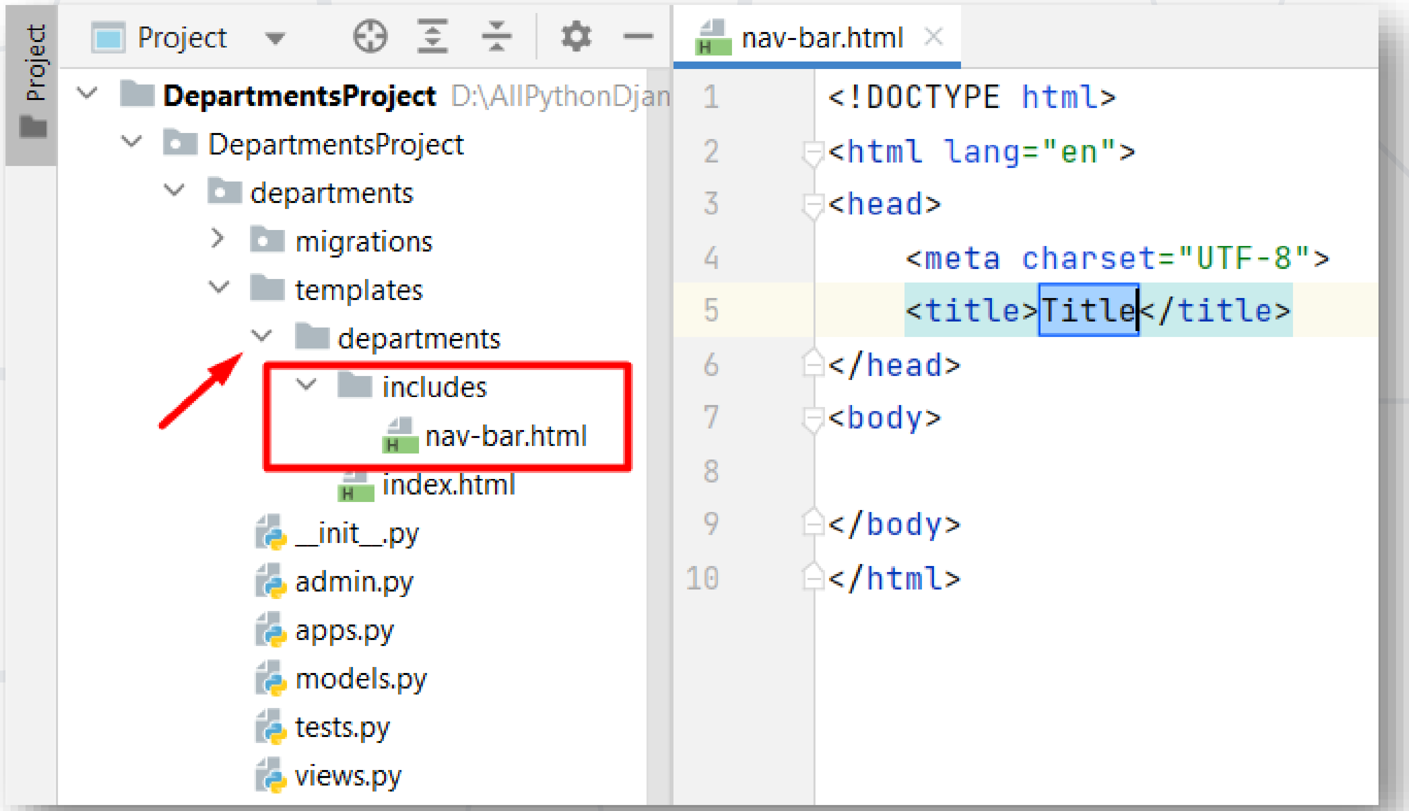
```
{% include template_name %}
```

  - a **quoted string** in single or double quotes

```
{% include "nav-bar.html" %}
```

Example: Including Template Snippets

- Create a new directory in the **app/templates/departments**
  - Create a **.html** file called **nav-bar**



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>

</body>
</html>
```

Example: Including Template Snippets

- Establish a **reusable** **navigation bar** logic that can be implemented across **various templates** as needed
- Consider including the code into the **base template**
  - to **enhance** **maintainability** and **consistency**

nav-bar.html

```
<header>
    <nav>
        {# some navigation bar logic #}
    </nav>
</header>
```

Template snippet

Example: Including Template Snippets

- **Include** the template **snippet**

```
index.html
```

```
{% extends "base.html" %}

{% block title %}Home Page{% endblock %}

{% block content %}
    {% include "departments/includes/nav-bar.html" %}
    {% for dep in departments.items %}
        <h2>{{ dep.title }}</h2>
        <p>{{ dep.description }}</p>
    {% endfor %}
{% endblock %}
```

Include with Context

- The included template **nav-bar.html** is **rendered within** the **context** of the template **index.html** that includes it
- Additional **context** can be **passed** to the included template using **keyword arguments**

```
{% include template_name with user='user_name' %}
```

- Alternatively, the included template can be **rendered** with **only** the **variables** provided or with **no variables** at all

```
{% include template_name with user='user_name' only %}
```

{% my_tag %}

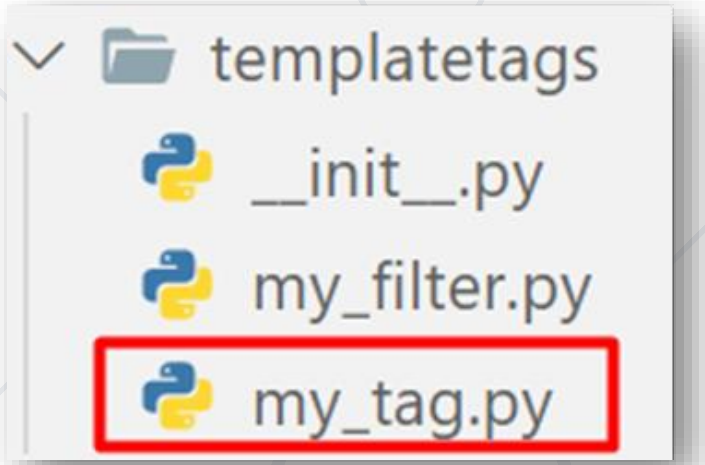
Custom Tags

Template Tags Helper Functions

- Django offers **helper functions** for creating **custom** template **tags**
  - **`simple_tag`**
    - Processes data and **returns** a **string**
  - **`inclusion_tag`**
    - Processes the data and **returns** a **rendered template**

Create templatetags Folder

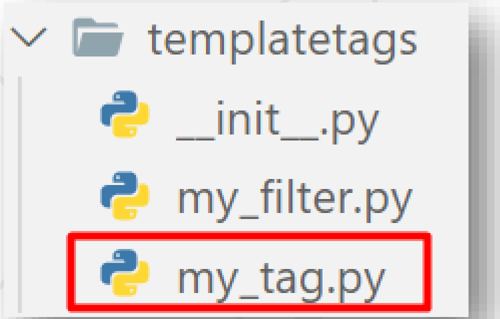
- In your application create a `templatetags` module
- Create a file `my_tag.py` in **templatetags** folder

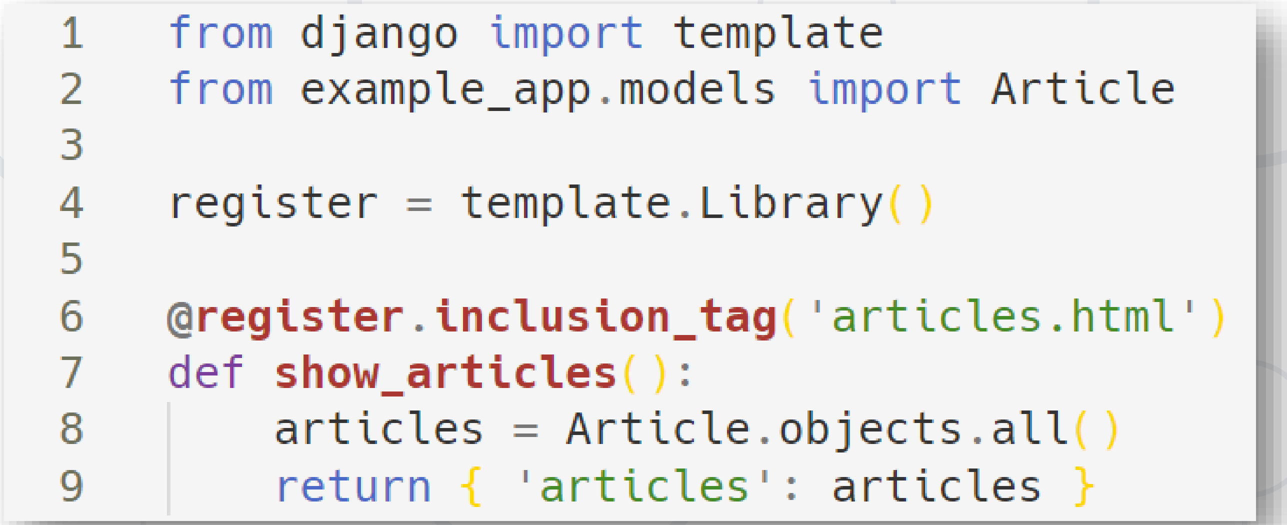


```
templatetags
    __init__.py
    my_filter.py
    my_tag.py
```

Creating Custom Template Tags

- Example of an **inclusion custom tag**





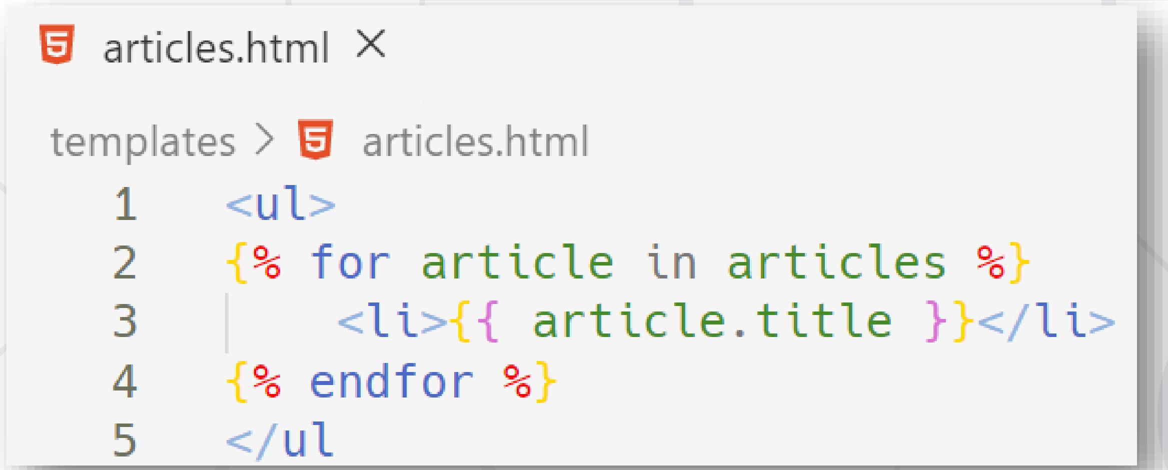
```
from django import template
from example_app.models import Article

register = template.Library()

@register.inclusion_tag('articles.html')
def show_articles():
    articles = Article.objects.all()
    return { 'articles': articles }
```

Using Custom Template Tags

- Create the **`articles.html`** template
  - **Loop** through the **articles**



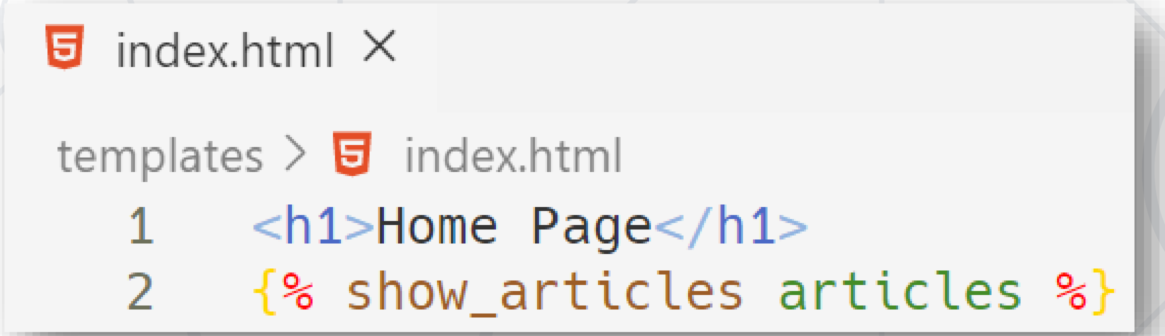
articles.html ×

templates > articles.html

```
<ul>
{% for article in articles %}
    <li>{{ article.title }}</li>
{% endfor %}
</ul
```

Using Custom Template Tags

- Use your tag in your **main** template (where you want to **include** the **list** of **articles**)
  - Load the **custom tags** at the top `{% load my_tag %}`
  - Use your **custom inclusion** tag `{% show_articles %}`



index.html

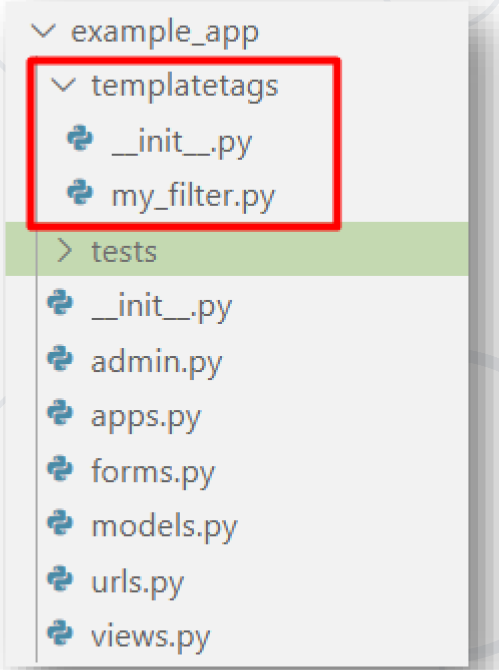
templates > index.html

```
<h1>Home Page</h1>
{% show_articles articles %}
```

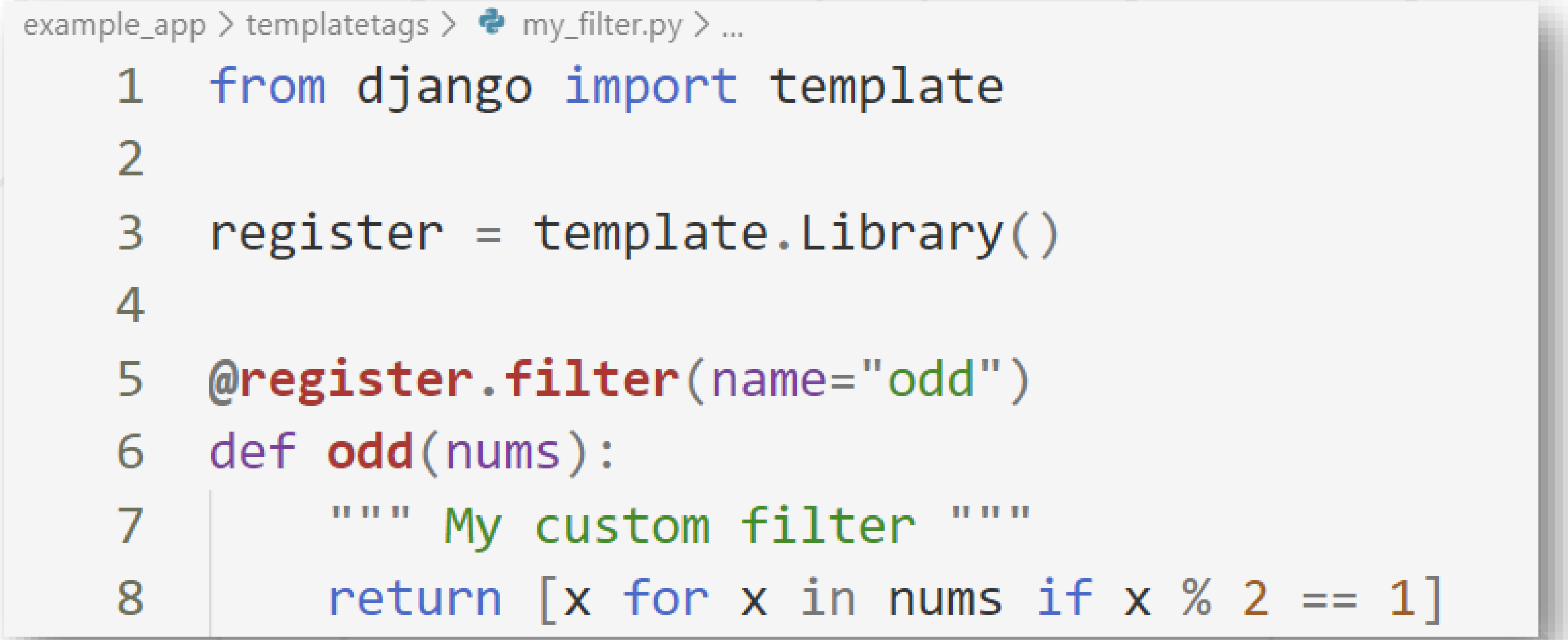
Custom Filters

Create a Custom Filter

- In the **templatetags** folder create a file **my_filter.py**
- Implement your **filter function**



```
example_app
  templatetags
    __init__.py
    my_filter.py
  tests
  __init__.py
  admin.py
  apps.py
  forms.py
  models.py
  urls.py
  views.py
```



example_app > templatetags > my_filter.py > ...

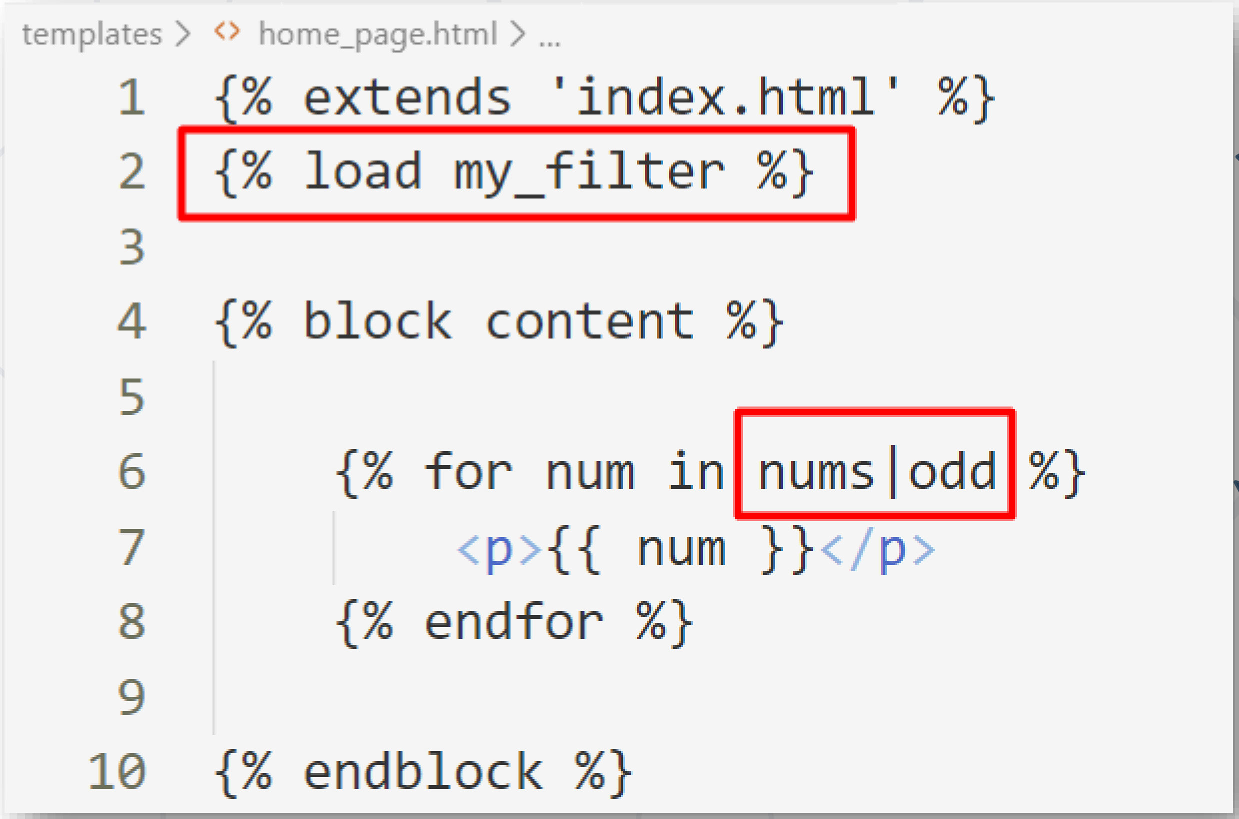
```python
from django import template

register = template.Library()

@register.filter(name="odd")
def odd(nums):
    """ My custom filter """
    return [x for x in nums if x % 2 == 1]
```

Use Your Custom Filter

- **Load** the **custom filters** in your template
- Use your **custom filter** named **odd**



templates > <> home_page.html > ...

```
{% extends 'index.html' %}
{% load my_filter %}

{% block content %}

    {% for num in nums|odd %}
        <p>{{ num }}</p>
    {% endfor %}

{% endblock %}
```

Use the name of your file

Use your custom filter name

Bootstrap

Front-end Framework

Bootstrap

- A popular **open-source** front-end **framework**
- **Simplifies** the process of **designing** and **building**
  - **responsive** and **mobile-first** web pages
- Includes a set of **pre-designed** HTML, CSS, and JavaScript **components**
- Read the Bootstrap 5 documentation

Bootstrap Stylesheet Link

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
crossorigin="anonymous">
    <title>Home Page</title>
</head>
```

Bootstrap Navbar

```html
<nav class="navbar navbar-expand-lg bg-body-tertiary">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">Navbar</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation"><span class="navbar-
toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav">
        <li class="nav-item">
          <a class="nav-link" href="#">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Features</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Pricing</a>
        </li>
     </ul>
    </div>
  </div>
</nav>
```

Bootstrap Cards

```html
<div class="card" style="width: 18rem;">
  <img src="https://cdn.pixabay.com/photo/2017/05/31/16/24/cat-
2360863_640.jpg" class="card-img-top" alt="Card image">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

Bootstrap Pros & Cons

- **Pros**
  - Rapid Development
  - Responsive Design
  - Cross-Browser Compatibility
  - Large Community
- **Cons**
  - Generic Look
  - Learning Curve
  - File Size
  - Overhead

Summary

- Template **Inheritance**
  - Block, Extend, Include
- **Custom** Template **Tags**
- **Custom Filters**
- **Bootstrap**
  - Front-end Framework

Software University

Questions?

SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

Finance Academy

SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SOFTWARE GROUP

DXC TECHNOLOGY

createX

Postbank

Решения за твоето утре

SUPER HOSTING .BG

DRAFT KINGS

VIVACOM

INDEAVR

Serving the high achievers

AMBITIONED

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, softuni.org
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License