

TRAINING DAY11 REPORT

05 JULY 2025

WHAT IS TEMPLATE INHERITANCE?

Template Inheritance in Django allows you to build a **base template** with common structure (like headers, footers, navigation bars, etc.) and then extend it in **child templates**, so you don't have to repeat the same code in every HTML file.

Why Template Inheritance?

Imagine you have a website with 10 pages. Each page shares the same header, footer, and sidebar. Instead of copying that HTML 10 times, you create one **base.html** and let other templates **inherit** from it.

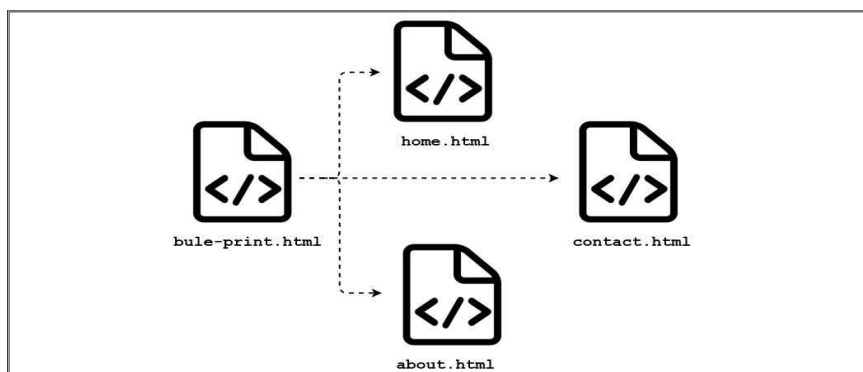
Core Syntax

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


Tells Django that this template is a child and extends the layout of base.html.

2. `{% block block_name %}` and `{% endblock %}`


Defines replaceable sections in the base template.



Example: Base Template

```
templates >  base.html > ...
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>{% block title %}My Website{% endblock %}</title>
5      <link rel="stylesheet" href="{% static 'css/style.css' %}">
6  </head>
7  <body>
8      <header>
9          <h1>My Website Header</h1>
10     </header>
11
12     <nav>
13         <!-- Navigation bar -->
14     </nav>
15
16     <main>
17         {% block content %}
18         <!-- Default content -->
19         {% endblock %}
20     </main>
21
22     <footer>
23         <p>My Website Footer</p>
24     </footer>
25 </body>
26 </html>
27
```

Child Template Example

```
templates >  home.html > ...
1  {% extends "base.html" %}
2
3  {% block title %}Home Page{% endblock %}
4
5  {% block content %}
6      <h2>Welcome to the Home Page</h2>
7      <p>This content is from the home.html file.</p>
8  {% endblock %}
9
```

How It Works:

- Django sees `{% extends "base.html" %}`.
- It loads `base.html`.
- It replaces any `{% block %}` in the base template with the ones in the child (if defined).

PIPE OPERATORS

In Django templates, a **pipe operator** (`|`) is used to apply **filters** to variables. Filters modify the value **before rendering it** in the HTML output.

Syntax:

```
{{ variable|filter_name }}
```

- The pipe operator passes the value of `variable` to the `filter_name`.
- You can also chain multiple filters:

```
{{ variable|filter1|filter2 }}
```

Example Use Cases:

1. Uppercase Text

```
{{ "hello world"|upper }}
```

Output: HELLO WORLD

2. Default Value

```
{{ user.username|default:"Guest" }}
```

If user.username is not available, it shows "Guest".

3. Date Formatting

```
{{ post.published|date:"F j, Y" }}
```

Output: July 5, 2025 (if the date is set)

4. Truncate a Long String

```
{{ long_text|truncatechars:20 }}
```

Output: This is a long str...

5. Chaining Filters

```
{{ name|lower|title }}
```

Converts to lowercase, then capitalizes each word.

Commonly Used Built-in Filters:

Filter	Description
upper	Converts string to uppercase
lower	Converts string to lowercase
title	Capitalizes each word
length	Returns the length of a list or string
date	Formats a date
truncatechars	Truncates string after x characters
default	Provides a fallback if variable is undefined
safe	Marks the string as safe (disables auto-escaping)
linebreaks	Converts newlines to <p>s
join	Joins a list using a string separator

Important Notes

- **Pipe operators only work in templates**, not in Python views or models.
- Filters **do not modify** the original data — they only affect the output display.
- You can **create custom filters** with `@register.filter` in a `templatetags` module.

Custom Filter Example:

Create a file in app:

```
blogapi > blog > 🐍 filters.py > ...
1  from django import template
2  register = template.Library()
3
4  @register.filter
5  ✓ def multiply(value, arg):
6    |     return value * arg
7
8
```

Load and use in template:

```
templates > 📄 home.html > ...
1  {% load myfilters %}
2  {{ 5|multiply:3 }}  {# Output: 15 #}
3
4
```