
Django Learning Report

Name   : Prajwal Bashyal
Institution: Bhaktapur Multiple Campus, BIT
Date   : 2025-12-19
Topic   : Django – Template Engine (DTL)
----

1. Django Template Engine

Django uses its own **Django Template Language (DTL)**.
Its syntax is **very similar to Jinja**, so many people casually call it *Jinja-style*.

Django templates allow:

- Displaying dynamic data
- Writing logic inside HTML
- Separating backend logic from frontend design

This follows the **MVC/MVT principle**:

- **Model** → Data
- **View** → Logic
- **Template** → Presentation (HTML)

---

2. Template Syntax Overview

Django templates mainly use **three types of tags**:

Syntax	Purpose
`{{ }}`	Display data (variables)
`{% %}`	Logic (conditions, loops, tags)
`{# #}`	Comments

---

3. Template Variables (`{{ }}`)

Template variables are used to **display data** sent from the Django view.

Example:

```html
<h1>{{ title }}</h1>
<p>{{ username }}</p>
```

These values come from the **context dictionary** in views.

---

4. Context & Dictionary (View ↔ Template Communication)

What is Context?

Context is a Python dictionary used to send data from a **view** to a **template**.

Why Context is Needed?

- Python code runs in views
- HTML cannot directly access Python variables
- Context acts as a **bridge**

View Example:

```python
def home(request):
    context = {
        'title': 'My Blog',
        'username': 'Prajwal',
        'posts': ['Post One', 'Post Two']
    }
    return render(request, 'home.html', context)
```

Template Example:

```html
<h1>{{ title }}</h1>
<p>Welcome, {{ username }}</p>
```

➡ **Key Point:**
Dictionary **keys** become **template variables**

---

5. Template Tags (`{% %}`)

Template tags control **logic and flow** inside templates.

5.1 `if` Tag (Condition)

html

```html
{% if user.is_authenticated %}
    <p>Welcome, {{ user.username }}</p>
{% else %}
    <p>Please login</p>
{% endif %}
```

Used for:

- Authentication checks
- Conditional display

5.2 `for` Loop Tag

Used to loop over lists, querysets, or arrays.

html

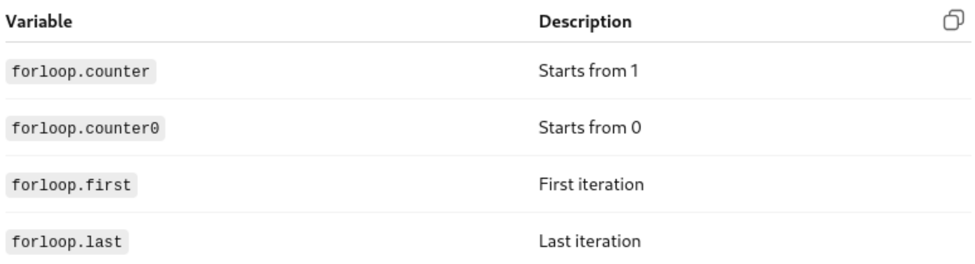
```html
<ul>
{% for post in posts %}
    <li>{{ post }}</li>
{% endfor %}
</ul>
```

Works with:

- Lists
- Django QuerySets
- Tuples

5.3 `forloop` Variables

Inside loops, Django provides built-in variables:



Variable	Description
`forloop.counter`	Starts from 1
`forloop.counter0`	Starts from 0
`forloop.first`	First iteration
`forloop.last`	Last iteration

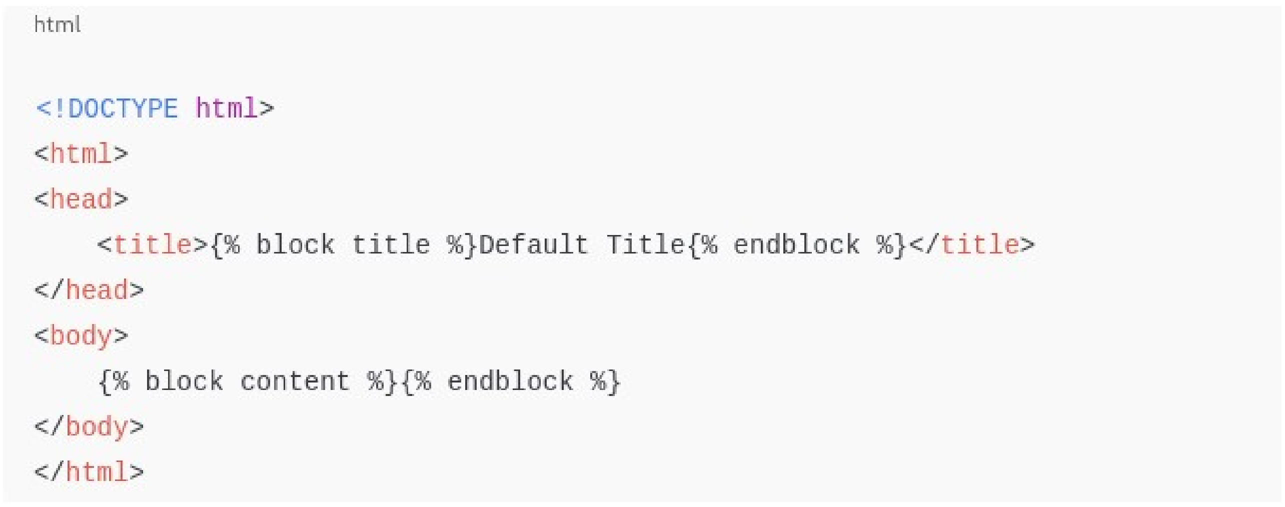
```
Example:

{{ forloop.counter }}. {{ post }}
```

6. `{% block %}` and `{% extends %}` (Template Inheritance)

Used to create **reusable layouts** .

Base Template (`base.html`)



html

```html
<!DOCTYPE html>
<html>
<head>
    <title>{% block title %}Default Title{% endblock %}</title>
</head>
<body>
    {% block content %}{% endblock %}
</body>
</html>
```

Child Template

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

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

{% block content %}
<h1>Welcome to Blog</h1>
{% endblock %}
```

➡ `Makes code` **`clean, DRY, and scalable`**

---

7. `{% include %}` Tag

Used to include reusable components.

```html
{% include 'navbar.html' %}
```

Useful for:

- Navbar
- Footer
- Sidebar

---

8. `{% url %}` Tag

Used to generate URLs dynamically.

```html
<a href="{% url 'post_detail' post.id %}">Read More</a>
```

Benefits:

- No hard-coded URLs
- Safe during URL changes

---

9. `{% load %}` Tag

Loads template libraries.

```html
{% load static %}
```

Required for:

- Static files
- Custom template filters

---

10. `{% static %}` Tag (Static Files)

```html
{% load static %}
<img src="{% static 'images/logo.png' %}">
```

Used for:

- CSS
- JS
- Images

---

11. Template Filters (|)

Filters modify variable output.

Common Filters:

Filter	Example	Output
`upper`	`{{ name	upper }}`
`lower`	`{{ name	lower }}`
`length`	`{{ posts	length }}`
`truncatewords`	`{{ text	truncatewords:5 }}`

12. Comments in Templates

```
{# This is a template comment #}
```

Not visible in browser source.

---

13. Summary

- Django templates separate **logic from design**
- **Context dictionary** sends data from views to templates
- `{{ }}` → Display data
- `{% %}` → Logic & control flow
- Template inheritance is essential for **blog websites**
- Template tags m