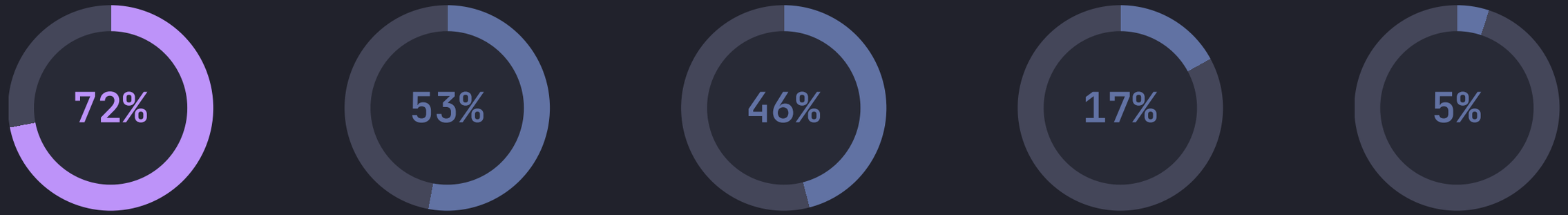
FULLSTACK NIGHTS · SEPTEMBER 2026

A Quick Look at Django Template Partials

Raúl Negrón-Otero
Senior Software Engineer, PostHog
https://raulnegron.me

Django Developers Survey 2026: how Django is used

WAYS RESPONDENTS SAID THEY USE DJANGO



Django Software Foundation and JetBrains PyCharm lp.jetbrains.com/django-developer-survey-2026

A super simple blog

THE TEMPLATE

entry.html

```
<h1>Why you should use Django</h1>
<p>by raul, 4 min read</p>
<p>Django 6.0 shipped ...</p>
<button>Like ({{ entry.likes }})</button>
```

IN THE BROWSER

Why you should use Django

by raul, 4 min read

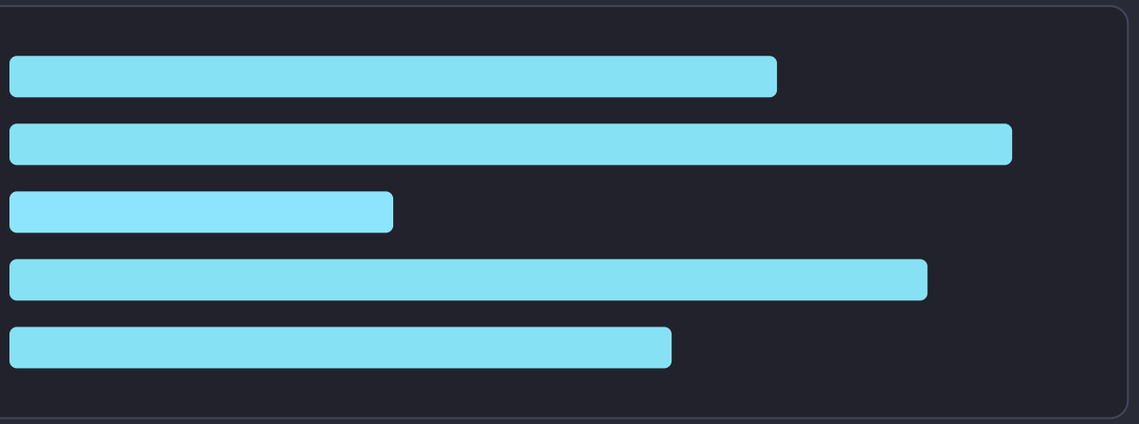
Django 6.0 shipped in December 2025.

Like (41)

- A blog entry with a **like button**
- Four lines of a **standard Django template**
- One click renders everything again just to change one number

Rebuilding the whole page, or just one piece

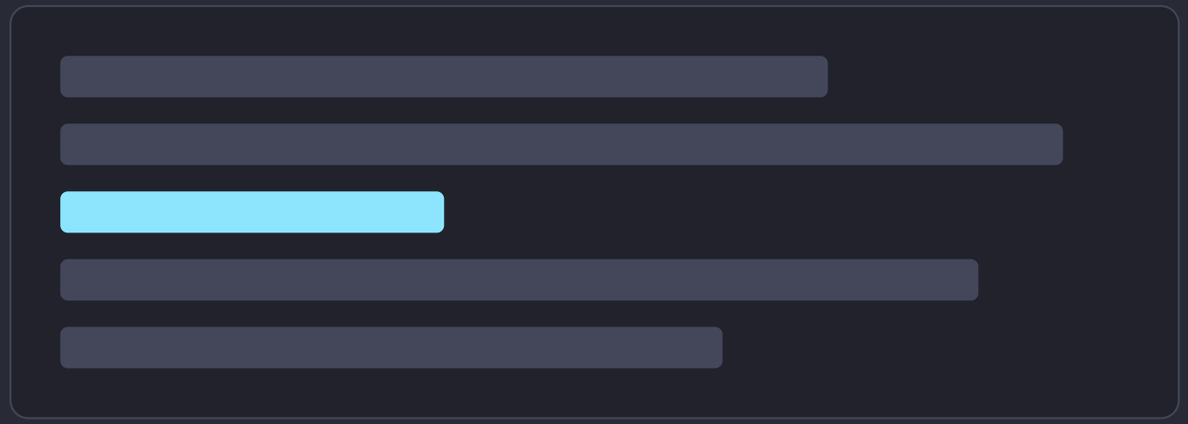
RELOAD THE WHOLE PAGE



response: **14 KB** of HTML

- The server renders the whole template
- The browser rebuilds every element

SWAP ONE FRAGMENT



response: **64 bytes** of HTML

- The server renders one fragment
- The browser replaces one element

Django and htmx: What are they?

DJANGO

- A **Python web framework**
- Traditionally sends a whole page per request

HTMX

- A **small JavaScript library**
- Lets HTML ask the server for more HTML

base.html

```
{% load static %}
<script src="{% static 'htmx.min.js' %}" defer></script>
```

views.py

```
def entry_detail(request, pk):
    entry = get_object_or_404(Entry, pk=pk)
    return render(request, "entry.html", {"entry": entry})
```

What htmx does when the button is clicked

entry.html

```
<button hx-post="/entry/1/like"
        hx-swap="outerHTML">
  Like ({{ entry.likes }})
</button>
```

- The reply is **HTML**, not JSON: the server sends the button back

Before: adding a fragment meant one more file

```
templates/
├─ entry.html
├─ _like.html
├─ _comments.html
└─ _related.html
```

- **include** and **block** both need a separate file
- The button is not in the file that uses it

entry.html

```
<h1>Why you should use Django</h1>
<p>by raul, 4 min read</p>
<p>Django 6.0 shipped ...</p>
{% include "_like.html" %}
```

_like.html

```
<button hx-post="/entry/1/like"
        hx-swap="outerHTML">
  Like ({{ entry.likes }})
</button>
```

Django 6.0 added template partials

entry.html

```
{% partialdef like_button %}
  <button>Like ({{ entry.likes }})</button>
{% endpartialdef %}

{% partial like_button %}
```

- **partialdef** names a fragment where it is used
- **partial** renders it, as often as needed
- Started as django-template-partials, now in core

Carlton Gibson: github.com/carltongibson/django-template-partials

Three ways to reuse a fragment of a page

include

Just pulls in another file

```
{% include "_like.html" %}
```

extends and block

Fills a block in a base template

```
{% extends "base.html" %}
{% block content %}
{% endblock %}
```

partialdef and partial

Names a fragment, then renders it

```
{% partialdef like_button %}
{% endpartialdef %}
{% partial like_button %}
```

A Django view can return just the button fragment

views.py

```python
def like(request, pk):
    entry = get_object_or_404(Entry, pk=pk)
    entry.likes += 1
    entry.save()
    return render(request, "entry.html#like_button", {"entry": entry})
```

- The **#like_button** suffix picks one fragment out of the template
- Works with **render**, **get_template** and **include**
- The response here is the HTML for the button on its own

The partial HTML can be used where it is defined

entry.html

```
{% partialdef like_button inline %}
  <button hx-post="/entry/1/like"
          hx-swap="outerHTML">
    Like ({{ entry.likes }})
  </button>
{% endpartialdef %}
```

- **inline** renders the fragment right where it is defined
- The htmx attributes are on the button itself
- One definition serves the first load and every update after

The original button, now a partial

Why you should use Django
by raul, 4 min read

Django 6.0 shipped in December 2025.

Like (41)

templates/
└ entry.html

what the server sent back

```
<button hx-post="/entry/1/like"
        hx-swap="outerHTML">
  Like (42)
</button>
```

- Only the button is replaced, nothing else moves
- Django rendered **just the button** and htmx swapped it in
- No page reload, no JavaScript written

A second partial in the same file

entry.html

```
<ul id="list">
  {% partialdef rows inline %}
    {% for c in comments %}
      <li>{{ c.author }}: {{ c.text }}</li>
    {% endfor %}
  {% endpartialdef %}
</ul>
<button hx-get="/entry/1/comments"
        hx-target="#list"
        hx-swap="beforeend">
  Load more
</button>
```

IN THE BROWSER

chris: cool!

raul: thanks!

Load more

- The view returns `rows`
- htmx appends them to the list

Files simplified: now we need just one

BEFORE

```
templates/
├─ entry.html
├─ _like.html
├─ _comments.html
└─ _related.html
```

entry.html

```
{% include "_like.html" %}
```

AFTER

```
templates/
└─ entry.html
```

entry.html

```
{% partialdef like_button inline %}
  <button hx-post="/entry/1/like"
          hx-swap="outerHTML">
    Like ({{ entry.likes }})
  </button>
{% endpartialdef %}
```

This idea may be heading into HTML itself

TRIPTYCH

- More HTTP methods directly from HTML
- Controls that can target any element
- A **WHATWG** proposal, polyfill available today

Alexander Petros github.com/alexpetros/triptych

DECLARATIVE PARTIAL UPDATES

- The browser streams HTML into named slots
- A **WICG** proposal, behind a flag in Chrome 148
- No cross-browser support yet

WICG github.com/WICG/declarative-partial-updates

A Quick Look at Django Template Partials

entry.html

```
{% partialdef like_button inline %}
  <button hx-post="/entry/1/like" hx-swap="outerHTML">
    Like ({{ entry.likes }})
  </button>
{% endpartialdef %}
```

views.py

```
def like(request, pk):
    entry = get_object_or_404(Entry, pk=pk)
    entry.likes += 1
    entry.save()
    return render(request, "entry.html#like_button", {"entry": entry})
```

Thanks!

Raúl Negrón-Otero
https://raulnegron.me

- Template partials, Django docs
  docs.djangoproject.com/en/6.0/ref/templates/builtins/
- django-template-partials, Carlton Gibson
  github.com/carltongibson/django-template-partials
- htmx, Big Sky Software
  htmx.org
- django-htmx, Adam Johnson
  github.com/adamchainz/django-htmx
- Triptych, Alexander Petros
  github.com/alexpetros/triptych
- Declarative Partial Updates, WICG
  github.com/WICG/declarative-partial-updates
- Django Developers Survey 2026, JetBrains
  lp.jetbrains.com/django-developer-survey-2026