

django

AULA 03

HERANÇA, COMPONENTES E CSS

O QUE VEREMOS HOJE

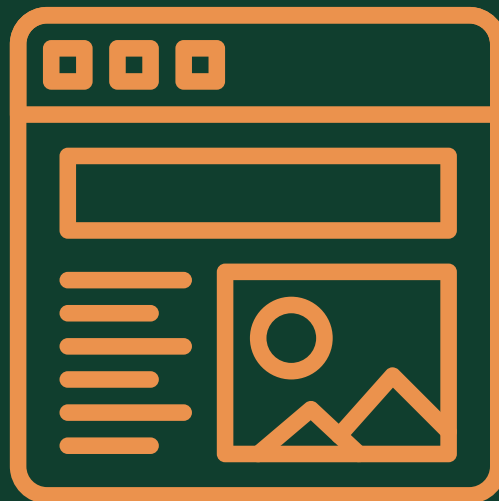
- 01 HERANÇA DE TEMPLATES
- 02 COMPONENTES
- 03 LINKS INTERNOS NO DJANGO
- 04 ESTILIZAÇÃO DE PÁGINAS COM CSS
- 05 APLICAÇÃO DO CSS NO DJANGO

HERANÇA DE TEMPLATES

Herança de templates é um conceito que permite criar uma estrutura base reutilizável para páginas web.

No Django, podemos criar um template "pai" que serve como esqueleto para várias páginas do site.

Ao usar a herança de templates, você pode definir a estrutura HTML comum a todas as páginas, como o cabeçalho, menu de navegação e rodapé.



criação de um template base

Para criar um template base no Django, é necessário primeiro definir a estrutura HTML básica da página, estabelecendo os blocos de conteúdo que serão preenchidos pelo conteúdo de outras páginas.

É possível definir blocos de conteúdo utilizando a tag `{% block %}`. Esses blocos podem ser preenchidos com conteúdo específico em cada página que herda do template base.


```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4     <title>{% block title %}Meu Site{% endblock %}</title>
5 </head>
6 <body>
7     <header>
8         <h1>Meu Site</h1>
9         <p>Bem-vindo ao meu site!</p>
10    </header>
11
12    <main>
13        {% block nome_do_bloco %}{% endblock %}
14    </main>
15
16    <footer>
17        <p>&copy; 2023 Meu Site. Todos os direitos reservados.</p>
18    </footer>
19 </body>
20 </html>
21
```

USO DO TEMPLATE EM UMA PÁGINA “FILHA”

Um template filho irá herdar do template base e sobrescrever apenas os blocos que precisam de conteúdo específico, mantendo a estrutura geral.

Para fazer isso precisamos indicar qual página está sendo usada como modelo através do extends.

E inserir o conteúdo específico da página dentro do bloco de conteúdo.



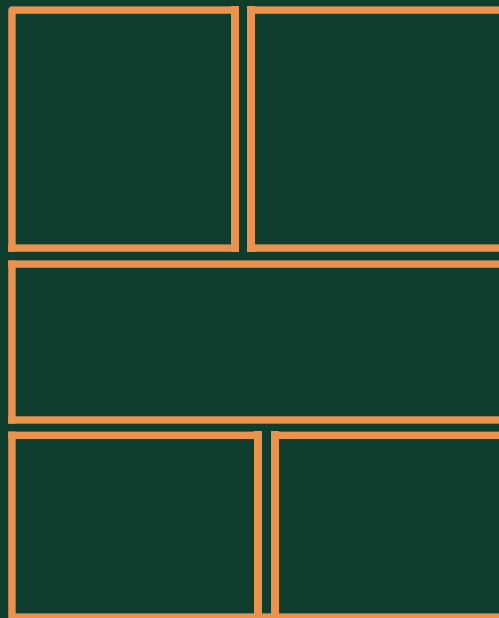
```
1 {% extends 'base.html' %}
2
3 {% block nome_do_bloco %}
4     <h1>Home</h1>
5     <p>Bem-vindo ao Meu site!</p>
6 {% endblock %}
```

COMPONENTIZAÇÃO DE TEMPLATES

Componentização é o ato de dividir uma estrutura maior (como um template completo) em partes menores e independentes, chamadas de componentes.

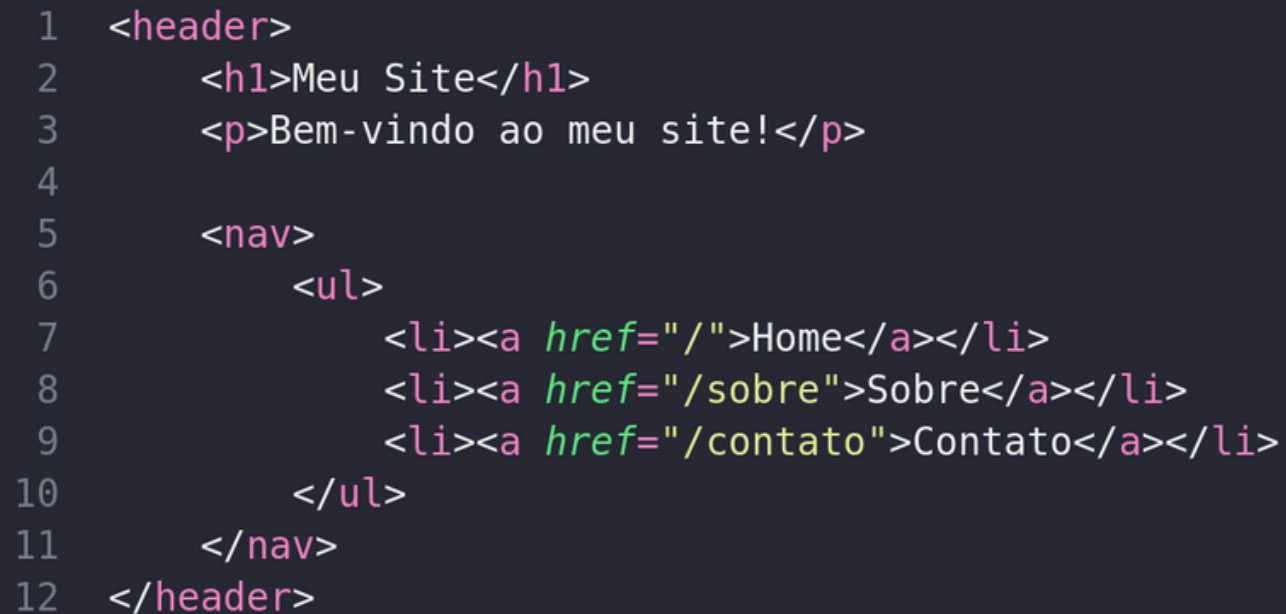
Cada componente tem uma função específica e pode ser reutilizado em diferentes partes do site.

Em Django, isso permite que o layout do site seja mais fácil de manter e organizar.



COMPONENTES NO DJANGO

Para criar um componente, só precisamos criar um arquivo html na pasta templates e adicionar apenas o conteúdo referente ao componente.



```
1  <header>
2      <h1>Meu Site</h1>
3      <p>Bem-vindo ao meu site!</p>
4
5      <nav>
6          <ul>
7              <li><a href="/">Home</a></li>
8              <li><a href="/sobre">Sobre</a></li>
9              <li><a href="/contato">Contato</a></li>
10         </ul>
11     </nav>
12 </header>
```

COMPONENTES NO DJANGO

E para utilizar o componente em um template, podemos incluir usando a tag `{% include %}`.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4     <title>{% block title %}Meu Site{% endblock %}</title>
5 </head>
6 <body>
7     {% include 'header.html'%}
8
9     <main>
10         {% block nome_do_bloco %}{% endblock %}
11     </main>
12
13     <footer>
14         <p>&copy; 2023 Meu Site. Todos os direitos reservados.</p>
15     </footer>
16 </body>
17 </html>
18
```


NAVEGAÇÃO ENTRE PÁGINAS

Para navegar entre páginas no HTML usamos a tag `<a>` que é usada para criar links.

Essa tag possui o atributo **href** que é usado pra definir o destino do link.

`Sobre`

Nesse exemplo, o link levará para página /sobre do mesmo site.



NAVEGAÇÃO ENTRE PÁGINAS NO DJANGO

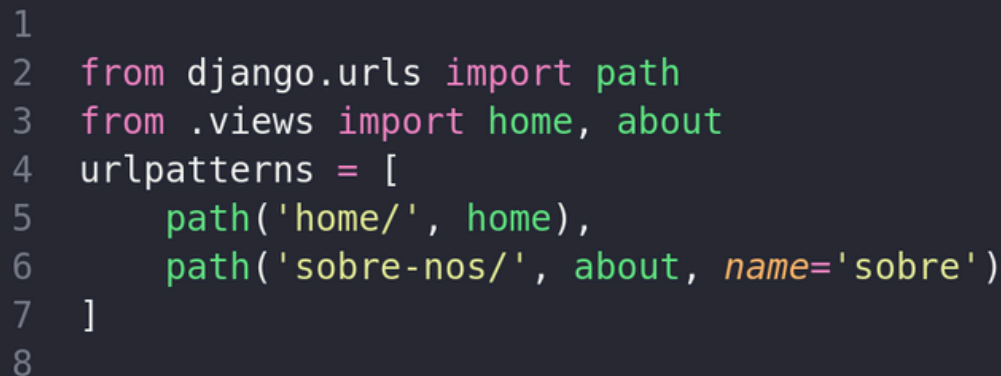
No Django, é possível substituir o uso direto de URLs na tag `<a>` pela tag de template `{% url %}`.

A tag `{% url %}` gera automaticamente o caminho para uma página, **baseada no nome da view** definido no arquivo `urls.py`.

Então no Django o mesmo link ficaria assim:

`Sobre`

E na url precisamos colocar o atributo `name` com o mesmo `sobre` passado na tag `<a>`

A code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains Python code for Django URL patterns.

```
1
2 from django.urls import path
3 from .views import home, about
4 urlpatterns = [
5     path('home/', home),
6     path('sobre-nos/', about, name='sobre')
7 ]
8
```

ATIVIDADE PRÁTICA

Crie duas páginas usando HTML no Django. Uma página vai exibir uma lista de produtos e outra vai exibir as categorias de produtos.

As páginas devem conter um header (com o link das duas páginas) e o conteúdo de cada uma delas como conteúdo principal.

Utilize herança para criar um template base que vai exibir o conteúdo das páginas e o include para o componente header.

Obs: Insira os produtos e categorias usando o loop for passando as informações pela view

CSS

Para estilizar a aparência de páginas HTML usamos o CSS (Cascading Style Sheets) que é uma linguagem de estilo utilizada para controlar a apresentação e o layout de elementos HTML em uma página web.

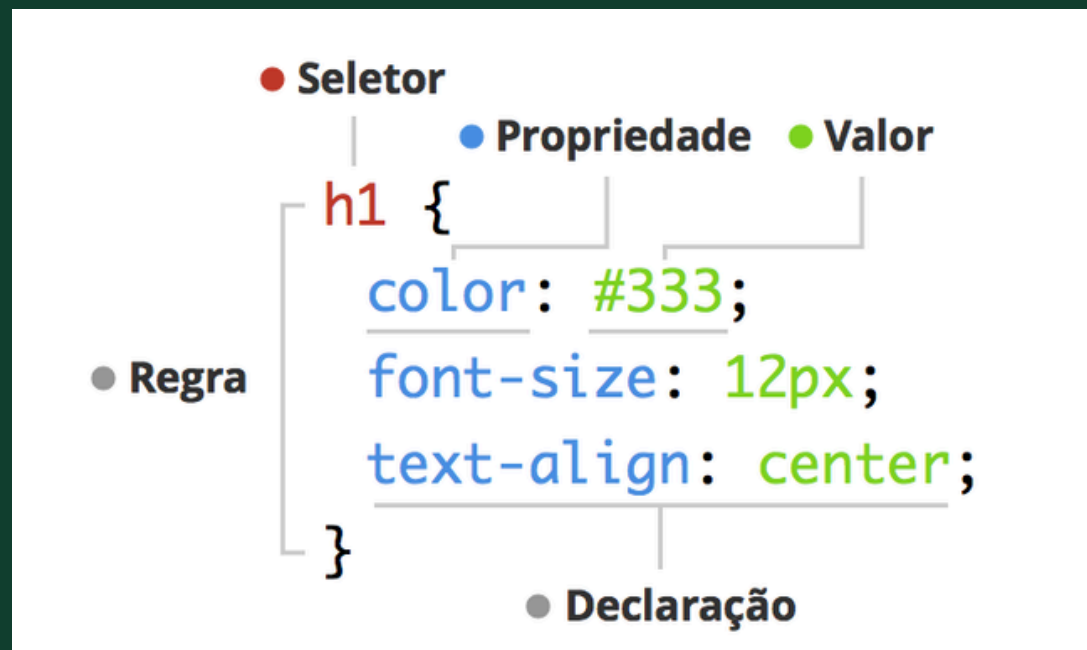
O CSS permite estilizar elementos HTML usando seletores, como elementos, classes, etc.



ESTRUTURA DO CSS

As regras do CSS são declaradas através de um seletor seguido por um bloco de declarações. Cada declaração consiste em uma propriedade seguida por um valor.

As propriedades CSS e seus respectivos valores, que são usados para estilizar elementos HTML. Isso inclui propriedades para cor, fonte, tamanho, margem, padding, entre outras.



CSS NO DJANGO

O Django já traz definido no `settings.py` uma pasta específica para adicionar CSS (e outros arquivos estáticos, como imagens e JavaScript) ao projeto.

Por conta disso, o ideal no Django é adicionar os arquivos CSS dentro da pasta **static**.



```
1 # Static files (CSS, JavaScript, Images)
2 # https://docs.djangoproject.com/en/5.1/howto/static-files/
3
4 STATIC_URL = 'static/'
```

CSS NO DJANGO

Uma vez criado os arquivos CSS, precisamos referenciar nos Templates para garantir que sejam aplicados.

Precisamos usar a tag `{% load static %}` para carregar arquivos estáticos nos templates.

E o `{% static 'css/style.css' %}` cria a URL para o arquivo `style.css` na pasta `static/css/`.

```
1 {% load static %}
2 <!DOCTYPE html>
3 <html lang="pt-br">
4 <head>
5     <title>Meu site</title>
6     <link rel="stylesheet" href="{% static 'css/style.css' %}">
7 </head>
8 <body>
9     {% include 'header.html'%}
10
11     <main>
12         {% block nome_do_bloco %}{% endblock %}
13     </main>
14
15     <footer>
16         <p>&copy; 2023 Meu Site. Todos os direitos reservados.</p>
17     </footer>
18 </body>
19 </html>
20
```

ATIVIDADE PRÁTICA

Aplique estilos nas páginas criadas no exercício anterior usando um arquivo CSS.

Para o header, defina uma cor leve e centralize o texto.

E para o conteúdo principal, aumente o tamanho da fonte e aumente o peso da fonte onde houver títulos.

Coloque um espaçamento entre os itens da lista.