

Reutilizando partes do template

Transcrição

Como resolveremos nosso problema de duplicação no código? Essa duplicação está no HTML, já que inserimos o `bootstrap.css` em dois lugares. Isso é um problema pois, caso quiséssemos colocar outra versão do Bootstrap ou o Bootstrap minificado, teríamos que alterar esses dois lugares. Se a quantidade de arquivos escalasse — por exemplo, para 20 ou 30 arquivos HTML —, teríamos que mudar todos eles, o que não é ideal.

O Flask tem uma solução de *templates* que nos ajuda a não precisar replicar tanto o código dessa maneira. Perceba que, além de `<link rel="stylesheet" href="../static/bootstrap.css">`, o `page-header` também é igual para as duas páginas, assim como `div class="container">`, `<body>`, `<head>`, as informações do título da página e o tipo de documento HTML.

Nós podemos pegar todo esse conteúdo e criar uma única página para contê-lo, criando uma nova página HTML chamada "template" e colando nela o conteúdo que é comum nas outras páginas.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Jogoteca</title>
    <link rel="stylesheet" href="../static/bootstrap.css">
  </head>
  <body>
    <div class="container">
      <div class="page-header">
        <h1>{{ titulo }}</h1>
      </div>
    </div>
  </body>
</html>
```

Nós não podemos simplesmente inserir essa página nas outras, por exemplo, por meio de herança. Na verdade podemos, mas precisaremos utilizar o Jinja2. A ideia é fazermos com que as outras páginas se encaixem logo após o título dessa página. Então, usaremos a diretiva do Flask (`{% %}`) para inserir um bloco de código, que chamaremos de conteúdo :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Jogoteca</title>
    <link rel="stylesheet" href="../static/bootstrap.css">
  </head>
  <body>
    <div class="container">
      <div class="page-header">
```

```

    <h1>{{ titulo }}</h1>
</div>

{% block conteudo %}{% endblock %}

</div>
</body>
</html>

```

Dessa forma, todo o nosso código HTML será herdado em volta do `conteudo` das nossas páginas. Vamos começar com a página `novo.html`. Nela, removeremos todo o código replicado. Para que a página `template.html` fique em volta do `conteudo` dessa, teremos que identificar a parte desejada do código como um bloco chamado `conteudo`, da mesma forma que fizemos no `template.html`.

```

{% block conteudo %}
    <form action="criar" method="post">
        <fieldset>
            <div class="form-group">
                <label for="nome">Nome</label>
                <input type="text" id="nome" name="nome" class="form-control">
            </div>
            <div class="form-group">
                <label for="categoria">Categoria</label>
                <input type="text" id="categoria" name="categoria" class="form-control">
            </div>
            <div class="form-group">
                <label for="console">Console</label>
                <input type="text" id="console" name="console" class="form-control">
            </div>
            <button type="submit" class="btn btn-primary btn-salvar">Salvar</button>
        </fieldset>
    </form>
{% endblock %}

```

Agora, precisamos herdar as informações do `template.html`. Para isso, utilizaremos `{% extends "" %}` e o nome do nosso template:

```

{% extends "template.html" %}
{% block conteudo %}
    <form action="criar" method="post">
        <fieldset>
            <div class="form-group">
                <label for="nome">Nome</label>
                <input type="text" id="nome" name="nome" class="form-control">
            </div>
            <div class="form-group">
                <label for="categoria">Categoria</label>
                <input type="text" id="categoria" name="categoria" class="form-control">
            </div>
            <div class="form-group">
                <label for="console">Console</label>
                <input type="text" id="console" name="console" class="form-control">
            </div>
            <button type="submit" class="btn btn-primary btn-salvar">Salvar</button>
        </fieldset>
    </form>

```

```
        </fieldset>
    </form>
{% endblock %}
```

Vamos testar? Acessando a URL <http://127.0.0.1:5000/novo> (<http://127.0.0.1:5000/novo>), veremos que nada mudou, que é exatamente o que esperávamos. Sendo assim, vamos repetir o processo para `lista.html` :

```
{% extends "template.html" %}
{% block conteudo %}
    <table class="table table-striped table-responsive table-bordered">
        <thead class="thead-default">
            <tr>
                <th>Nome</th>
                <th>Categoria</th>
                <th>Console</th>
            </tr>
        </thead>
        <tbody>
            {% for jogo in jogos %}
                <tr>
                    <td>{{ jogo.nome }}</td>
                    <td>{{ jogo.categoria }}</td>
                    <td>{{ jogo.console }}</td>
                </tr>
            {% endfor %}
        </tbody>
    </table>
{% endblock %}
```

Da mesma forma, a URL <http://127.0.0.1:5000/> (<http://127.0.0.1:5000/>) continuará funcionando perfeitamente. Existem alguns problemas que ainda não estão claros, mas falaremos sobre eles quando abordarmos boas práticas de criação de templates.