Spring Boot +Thymeleaf Template

Herysson R. Figueiredo herysson.figueiredo@ufn.edu.br

O primeiro passo é criarmos o nosso template base, é esse template que irá conter toda a estrutura base que os outros templates irão herdar, ou seja tudo aquilo que é comum em todas as páginas de nossa aplicação.

```
<!DOCTYPE html>
<html lang="pt-br" xmlns:th="http://www.w3.org/1999/xhtml" th:fragment="layout (conteudo)">
    <head>
        <meta charset="UTF-8">
        <title>CRUD SITE</title>
    </head>
    <body>
        <header>
            <h1>MINHA NAV BAR</h1>
        </header>
        <main th:include="${conteudo}">
            Contéudo Principal
        </main>
        <footer>
            <span>MEU SUPER RODAPÉ</span>
        </footer>
    </body>
</html>
```

O que estamos fazendo no código mostrado é criando um fragment chamado *layout* que recebe um parâmetro chamado **conteudo**, perceba que o valor do parâmetro **conteudo** está sendo colocado na *tag main* através do atributo *th:include*, ou seja, o valor esperado pelo parâmetro **conteudo** é um outro fragment e isso é perfeitamente possível de ser feito com o Thymeleaf.

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.w3.org/1999/xhtml"</pre>
                                                 th:replace="~{template :: layout (~{:: main})}">
<body>
   <main>
       <form action="#" th:action="@{/pessoa/save}" th:object="${pessoa}" method="post">
          Nome*:
                 <input type="text" th:field="*{nome}" required/>
             E-mail*:
                 <input type="text" th:field="*{email}" required/>
```

O que está acontecendo acima é que estamos dizendo que todo o conteúdo HTML de nossos templates filhos serão substituídos pelo conteúdo do fragment *layout* definido em nosso template **template.html**, porém estamos passando para o parâmetro do fragment *layout* toda a nossa tag *main*.

Veja que durante a passagem do parâmetro estamos utilizando a sintaxe do Thymeleaf quando trabalhamos com fragments, o que você pode estar estranhando é que estamos passando um fragment chamando main mas não estamos informando o local onde o fragment está localizado, o que acontece é que quando omitimos o arquivo onde se localiza o fragment, o Thymeleaf busca por tal fragment no arquivo de template atual e também não é necessário definirmos a tag main como um fragment, pois é possível informar uma tag diretamente, como estamos fazendo nos exemplos acima.

Bulma é uma estrutura de código aberto gratuita que fornece componentes de front-end prontos para uso que você pode combinar facilmente para criar interfaces da Web responsivas.

É uma biblioteca CSS. Isso significa que ele fornece classes CSS para ajudá-lo a estilizar seu código HTML.

https://bulma.io/

Bulma é uma estrutura de código aberto gratuita que fornece componentes de front-end prontos para uso que você pode combinar facilmente para criar interfaces da Web responsivas.









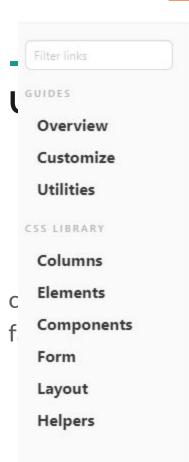






Download

Sponsor



Documentation

Everything you need to create a website with Bulma



An overview of what Bulma as a framework is all about







Eν









Download

Sponsor



Documentation

W Start You only need 1 CSS file to use Bulma

Overview

An overview of what Bulma as a framework is all about

Components Breadcrumb Card Dropdown Menu Message Modal Navbar Pagination Panel Tabs

of possibilities

· Breadcrumb

A simple **breadcrumb** component to improve your navigation experience



An all-around flexible and composable component

Basic Navbar

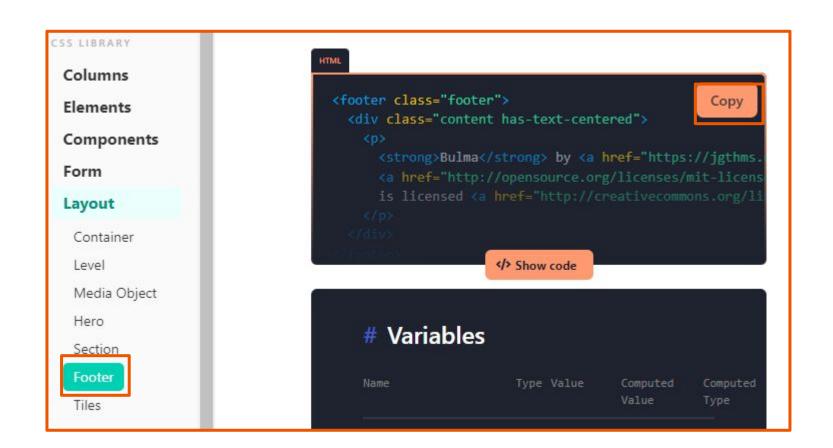
To **get started quickly**, here is what a complete basic navbar looks like:

Utilizan



```
<nav class="navbar" role="navigation" aria-label="main n"</pre>
 <div class="navbar-brand">
    <a class="navbar-item" href="https://bulma.io">
      <img src="https://bulma.io/images/bulma-logo.png" width="112</pre>
   <a role="button" class="navbar-burger" aria-label="menu" aria-</p>
      <span aria-hidden="true"></span>
      <span aria-hidden="true"></span>
      <span aria-hidden="true"></span>
 <div id="navbarBasicExample" class="navbar-menu">
    <div class="navbar-start">
```

```
<!DOCTYPE html>
<html lang="pt-br" xmlns:th="http://www.w3.org/1999/xhtml" th:fragment="layout (conteudo)">
    <head...>
    <body>
        <header...>
        <main th:include="${conteudo}">
            Contéudo Principal
        </main>
       <footer class="footer"...>
    </body>
</html>
```



```
<!DOCTYPE html>
<html lang="pt-br" xmlns:th="http://www.w3.org/1999/xhtml" th:fragment="layout (conteudo)">
    <head...>
    <body>
        <header...>
        <main th:include="${conteudo}">
            Contéudo Principal
        </main>
        <footer class="footer"...>
    </body>
</html>
```

Utilizar

```
<body>
   <header...>
    <section class="hero is-dark is-fullheight">
        <div class="hero-body">
            <div class="container">
                <div class="columns is-centered">
                    <div class="column is-half">
                        <main th:include="${conteudo}">
                            Contéudo Principal
                        </main>
                    </div>
                </div>
            </div>
        </div>
   </section>
   <footer class="footer"...>
</body>
```

Modifique o estilo da tabela

Modifique Centralize o titulo

Typography helpers

Modifique o estilo do titulo

<h2 class="subtitle">Subtitle</h2>

JPA - Hibernate Association

https://www.baeldung.com/jpa-hibernate-associations

https://www.baeldung.com/hibernate-one-to-many

Bibliografia

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. UML: guia do usuário. 2. ed. Rio de Janeiro: Campus, 2006.

PRESSMAN, Roger S. Engenharia de software. 5. Ed. Rio de Janeiro: McGraw-Hill, 2002.

SOMMERVILLE, I. Engenharia de software. 8. Ed. São Paulo, SP: Addison Wesley, 2007

BEZERRA, Eduardo. Princípios de Análise e Projeto de Sistemas com UML. Rio de Janeiro: Campus, 2006.