03

Aplicando classes

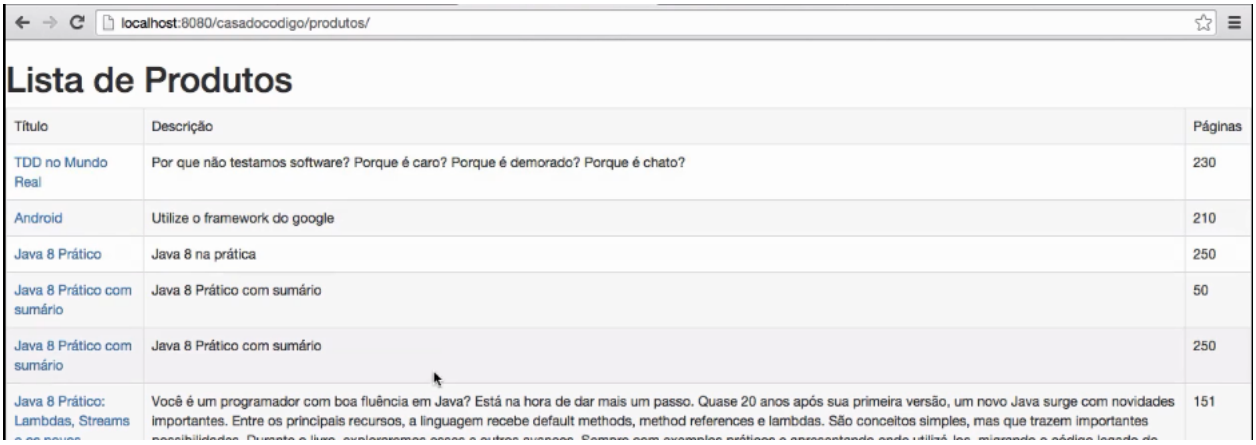
Transcrição

Com o *Bootstrap* configurado no projeto, podemos começar a usar as classes do mesmo para estilisar as páginas da aplicação. Comecemos pela listagem dos produtos no arquivo `lista.jsp`.

Se verificarmos a documentação do próprio *Bootstrap*, ela nos forcene uma série de exemplos de como suas classes. Explore a documentação em getbootstrap.com/css (http://getbootstrap.com/css). Na seção de tabelas, encontra-se uma boa variedade de estilização de tabelas. Para a listagem dos produtos, experimentamos as seguintes classes: `table table-bordered table-striped table-hover`.

Lembre-se de que estas classes são adicionadas na tag `table` da `lista.jsp`.

```
<table class="table table-bordered table-striped table-hover">
    <tr>
        <td>Título</td>
        <td>Descrição</td>
        <td>Páginas</td>
    </tr>
    [...]
</table>
```



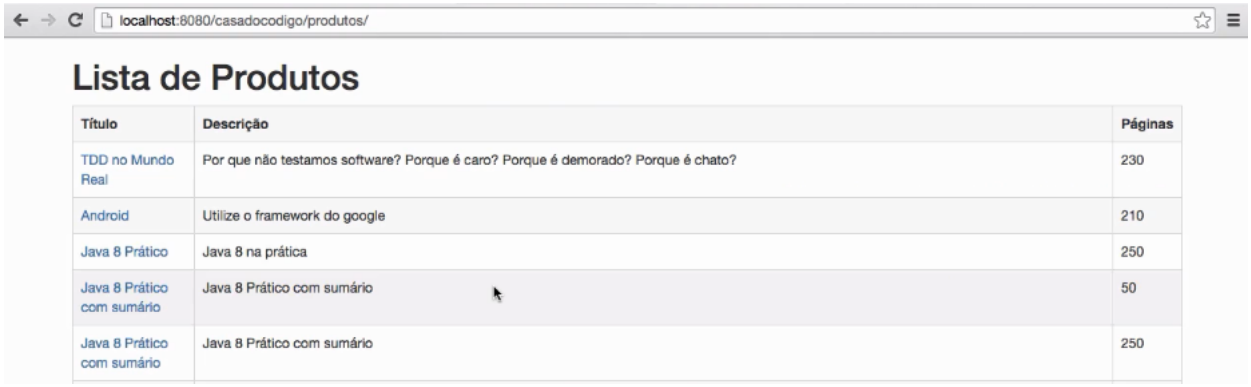
Uma pequena modificação possível no HTML é trocar as tags de título do cabeçalho da tabela de `td` para `th`. Visualmente, os títulos ficaram em negrito, mas o HTML também ficará um pouco mais semântico, agora que os cabeçalhos são realmente títulos em vez de texto solto em colunas.

```
<table class="table table-bordered table-striped table-hover">
    <tr>
        <th>Título</th>
        <th>Descrição</th>
        <th>Páginas</th>
    </tr>
    [...]
</table>
```

Uma outra boa modificação é envolver a listagem dos produtos com uma `div` que usa a classe `container` do *Bootstrap*. Ela centraliza os elementos na página, não deixando que tudo fique muito nos cantos da tela. Assim teremos:

```
<div class="container">
    <h1>Lista de Produtos</h1>
    <p> ${sucesso} </p>
    <p> ${falha} </p>
    <table class="table table-bordered table-striped table-hover">
        <tr>
            <th>Título</th>
            <th>Descrição</th>
            <th>Páginas</th>
        </tr>
        [...]
    </table>
</div>
```

E visualmente, a página fica mais agradável de ser vista e mais confortável de ser lida:



Com estas estilizações prontas, podemos começar a estilizar um pouco o formulário de cadastro. Mas observe o seguinte: Toda vez que queremos ir de uma página para outra, temos que digitar a url da página na barra de endereços do navegador. Vamos criar um menu que facilite essa navegação entre as páginas. Deixamos o HTML deste menu prontinho para usar, veja o trecho de código abaixo:

```
<nav class="navbar navbar-inverse">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target='
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">Casa do Código</a>
    </div>
    <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
      <ul class="nav navbar-nav">
        <li><a href="#">Lista de Produtos</a></li>
        <li><a href="#">Cadastro de Produtos</a></li>
      </ul>
    </div><!-- /.navbar-collapse -->
  </div>
</nav>
```

O código acima cria um menu de navegação usando as classes do próprio *Bootstrap*. Copie-o para o projeto. Perceba que há dois links no menu que são para as páginas de listagem e cadastro de produtos. Estes links ainda não funcionam, precisamos criá-los através do `s:mvcUrl` . Assim teremos:

```
<ul class="nav navbar-nav">
    <li><a href="${s:mvcUrl('PC#listar').build()}">Lista de Produtos</a></li>
    <li><a href="${s:mvcUrl('PC#form').build()}">Cadastro de Produtos</a></li>
</ul>
```

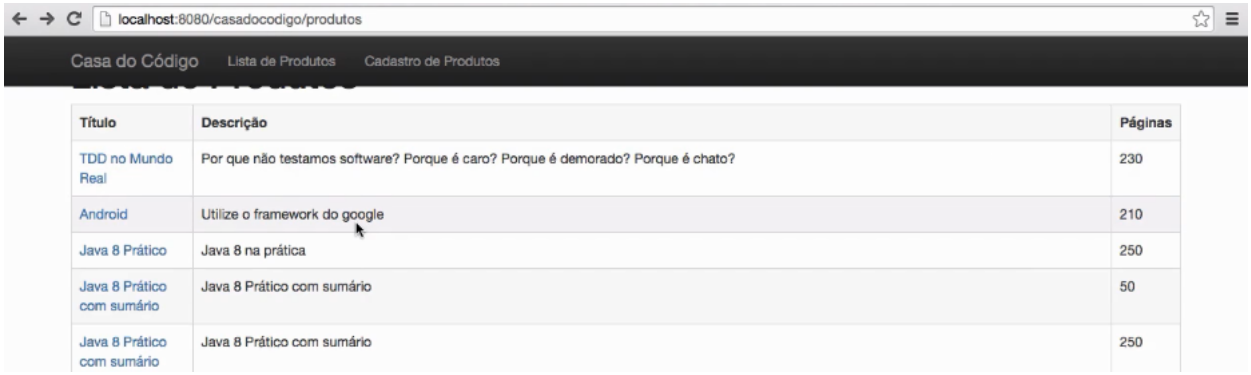
Nenhuma novidade até aqui. O código do menu completo fica da seguinte forma:

```
<nav class="navbar navbar-inverse">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target='
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">Casa do Código</a>
    </div>
    <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
      <ul class="nav navbar-nav">
        <li><a href="${s:mvcUrl('PC#listar').build()}">Lista de Produtos</a></li>
        <li><a href="${s:mvcUrl('PC#form').build()}">Cadastro de Produtos</a></li>
    </ul>
    </div><!-- /.navbar-collapse -->
  </div>
</nav>
```

Este código deve ficar acima da listagem, desta forma o menu ficará fixo no topo da página. Teremos no código algo como:

```
<body>
<nav class="navbar navbar-inverse">
    [...]
</nav>
<div class="container">
    <h1>Lista de Produtos</h1>
    <p> ${message} </p>
    <table class="table table-bordered table-striped table-hover">
    [...]
    </table>
</div>
</body>
```

E visualmente:



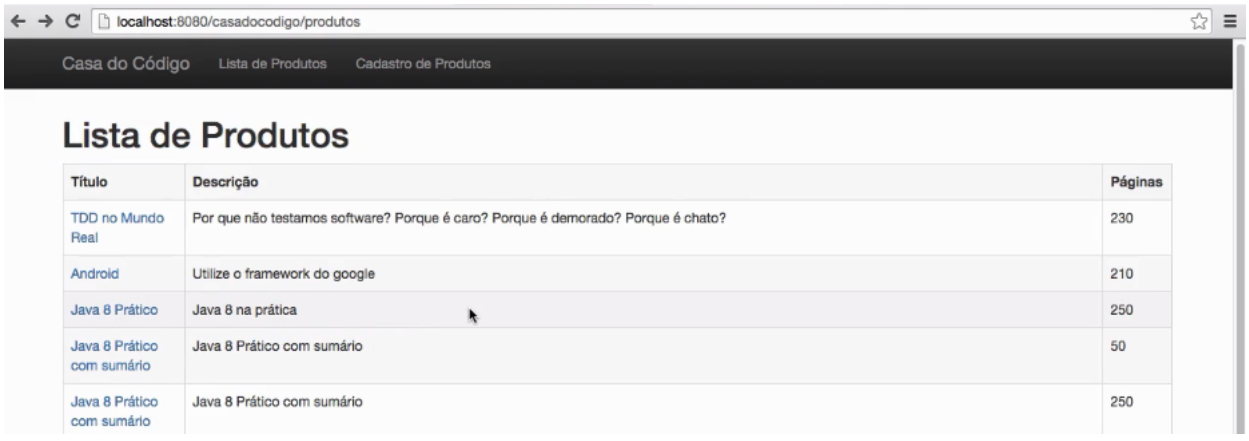
Note que o menu ficou sobrepondo o título da página. Corrigiremos este problema da forma mais simples. Usaremos a tag `style` dentro da tag `head`. Em seguida, modificaremos a propriedade `padding-top` da tag `body`, adicionando uma margem de `60px`. Desta forma o título da página será posicionado mais para baixo do menu. No código teremos:

```
<head>
<meta charset="UTF-8">
<title>Livros de Java, Android, iPhone, Ruby, PHP e muito mais - Casa do Código</title>
<link rel="stylesheet" href="resources/css/bootstrap.min.css" />
<link rel="stylesheet" href="resources/css/bootstrap-theme.min.css" />

    <style type="text/css">
        body{
            padding-top: 60px;
        }
    </style>

</head>
```

Atualizando a página teremos:



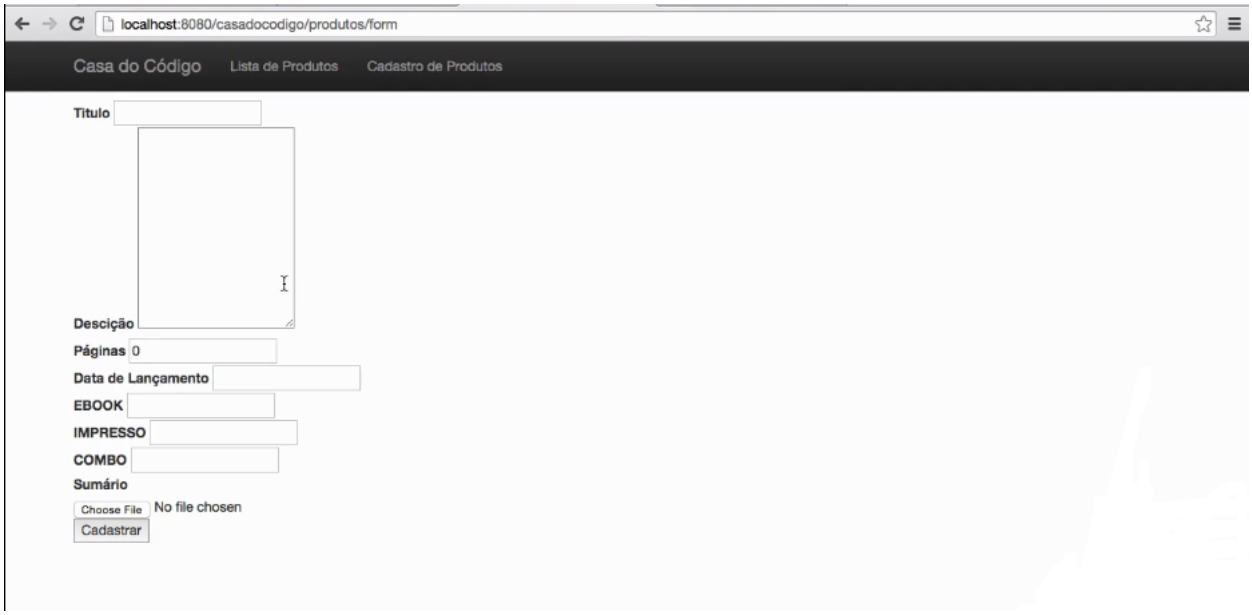
A página já está bem estilizada. Precisaremos estilizar também a página de cadastro de produtos. Vamos copiar todo o código do menu junto com a `div` que tem a classe `container` e colar no `form.jsp`. Não podemos esquecer de copiar o código `css` que está no `head` também. Assim teremos na página `form.jsp` o seguinte código:

```
<head>
    <meta charset="UTF-8">
    <title>Livros de Java, Android, iPhone, Ruby, PHP e muito mais - Casa do Código</title>
    <c:url value="/resources/css" var="cssPath" />
    <link rel="stylesheet" href="${cssPath}/bootstrap.min.css" />
    <link rel="stylesheet" href="${cssPath}/bootstrap-theme.min.css" />
```

```
    <style type="text/css">
        body{
            padding-top: 60px;
        }
    </style>
</head>
<body>
    <nav class="navbar navbar-inverse">
      [...]
    </nav>
    <div class="container">

        <form:form action="${s:mvcUrl('PC#gravar').build() }" method="post" commandName="produto
            [...]
        </form:form>
    </div>
</body>
```

A página de cadastro já parece bem mais atraente:



A estilização de formulários com o *Bootstrap* é bem simples. Note que todos os campos estão envolvidos por um `div` e que a maioria dos campos são criados com as tags do próprio *Spring*, como por exemplo `<form:input>`. Nestes casos, adicionaremos a classe `form-group` em cada um dos `div` s e o atributo `cssClass` com o valor `form-control` em cada um dos `<form:input>`. Até este ponto teremos o seguinte formulário:

```
<form:form action="${s:mvcUrl('PC#gravar').build() }" method="post" commandName="produto" encty
    <div class="form-group">
        <label>Título</label>
        <form:input path="titulo" cssClass="form-control" />
        <form:errors path="titulo" />
    </div>
    <div class="form-group">
        <label>Descrição</label>
        <form:textarea rows="10" cols="20" path="descricao" cssClass="form-control" />
        <form:errors path="descricao" />
    </div>
    <div class="form-group">
        <label>Páginas</label>
```

```
            <form:input path="paginas" />
            <form:errors path="paginas" />
        </div>
        <div class="form-group">
            <label>Data de Lançamento</label>
            <form:input path="dataLancamento" cssClass="form-control" />
            <form:errors path="dataLancamento" />
        </div>
        <c:forEach items="${tipos}" var="tipoPreco" varStatus="status">
            <div class="form-group">
                <label>${tipoPreco}</label>
                <form:input path="precos[${status.index}].valor" cssClass="form-control" />
                <form:hidden path="precos[${status.index}].tipo" value="${tipoPreco}" />
            </div>
        </c:forEach>

        <div class="form-group">
            <label>Sumário</label>
            <input name="sumario" type="file" />
        </div>
        <button type="submit">Cadastrar</button>
    </form:form>
```

Nas tags que não foram criadas com o `form:input` do *Spring* usaremos o atributo `class` do próprio HTML. Veja o `input file` por exemplo:

```
    <div class="form-group">
        <label>Sumário</label>
        <input name="sumario" type="file" class="form-control" />
    </div>
```

Como o *Bootstrap* já estiliza bem os campos do formulário, vamos remover os atributos `cols` e `rows` do campo `<form:textarea>`. Vamos também adicionar as classes `btn btn-primary` no botão de submissão do formulário. Também adicionaremos um título antes do formulário com o seguinte texto: `Cadastro de Produto`. Observe estas mudanças no código:

```
    <h1>Cadastro de Produto</h1>
    <form:form action="${s:mvcUrl('PC#gravar').build() }" method="post" commandName="produto" encty
        <div class="form-group">
            <label>Título</label>
            <form:input path="titulo" cssClass="form-control" />
            <form:errors path="titulo" />
        </div>
        <div class="form-group">
            <label>Descrição</label>
            <form:textarea path="descricao" cssClass="form-control" />
            <form:errors path="descricao" />
        </div>
        <div class="form-group">
            <label>Páginas</label>
            <form:input path="paginas" />
            <form:errors path="paginas" />
        </div>
```

```
    <div class="form-group">
        <label>Data de Lançamento</label>
        <form:input path="dataLancamento" cssClass="form-control" />
        <form:errors path="dataLancamento" />
    </div>
    <c:forEach items="${tipos}" var="tipoPreco" varStatus="status">
        <div class="form-group">
            <label>${tipoPreco}</label>
            <form:input path="precos[${status.index}].valor" cssClass="form-control" />
            <form:hidden path="precos[${status.index}].tipo" value="${tipoPreco}" />
        </div>
    </c:forEach>

    <div class="form-group">
        <label>Sumário</label>
        <input name="sumario" type="file" class="form-control" />
    </div>
    <button type="submit" class="btn btn-primary">Cadastrar</button>
</form:form>
```

Agora se atualizarmos a página, veremos que no final da página, os campos parecem estar bem juntos ao final da mesma. Para aumentar um pouco deste espaço, vamos alterar o `padding` que está sendo usando no código `css` dentro da tag `head` para não espaçar somente o topo da página, mas também sua parte inferior:

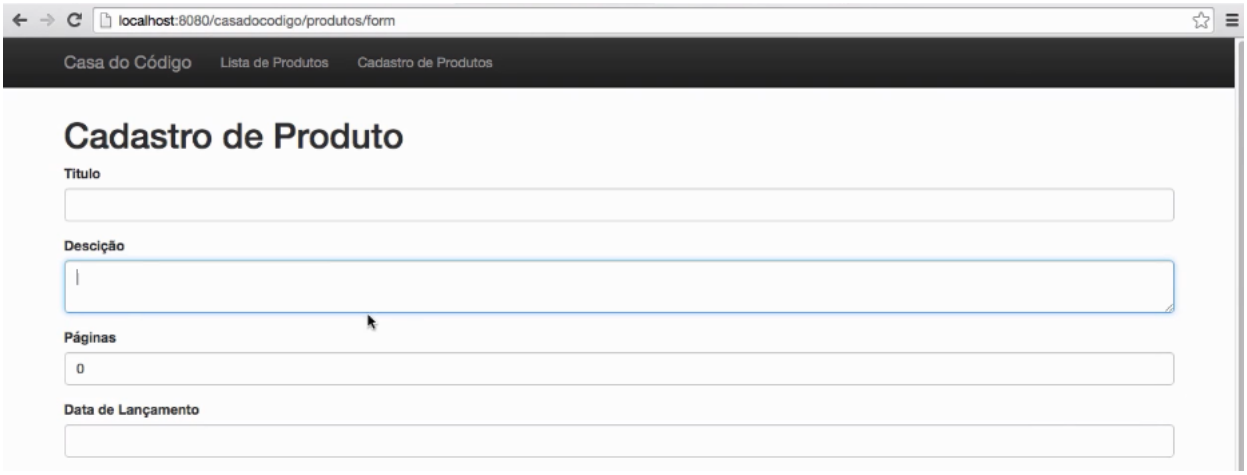
O código que estava assim:

```
<head>
    [...]
    <style type="text/css">
        body{
            padding-top: 60px;
        }
    </style>
</head>
```

Ficará da seguinte forma:

```
<head>
    [...]
    <style type="text/css">
        body{
            padding: 60px 0px;
        }
    </style>
</head>
```

Assim o espaçamento é aplicado tanto na parte superior, quanto na inferior da página. O visual final do formulário é bem agradável. Verifique como ficou.



Por último, colocaremos um link no menu de navegação que nos levará para a página inicial da aplicação. Na verdade, o link já foi criado, só precisaremos fazê-lo funcionar. Procure pelo seguinte trecho de código no menu:

```
<nav class="navbar navbar-inverse">
    <div class="container">
       <div class="navbar-header">
             <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data
       [...]
             </button>
             <a class="navbar-brand" href="#">Casa do Código</a>
              </div>
     </div>
</nav>
```

Observe a seguinte linha deste treho:

```
<a class="navbar-brand" href="#">Casa do Código</a>
```

Lembra como fazer para o *Spring* gerar o link correto? Fácil, usando a tag `s:mvcUrl` . Assim teremos:

```
<a class="navbar-brand" href="${s:mvcUrl('HC#index').build()}">Casa do Código</a>
```

Lembre-se de copiar esta linha para o menu do arquivo `lista.jsp` . Por ora, é bem desconfortável ficar copiando e colando códigos de um lado para o outro, mas futuramente veremos uma forma de melhorar isso.

Até aqui, aprendemos uma série de coisas nesta aula. Configuramos o *Bootstrap* em nosso projeto, configuramos para que o *Spring* não tente carregar os arquivos de estilo e script. Aprendemos a usar alguns estilos do *Bootstrap* e melhoramos bastante o visual das páginas de listagem e cadastro de produtos.

Caso tenha dúvidas sobre como usar o *Bootstrap*, experimente explorar a documentação e fazer também o Curso de Bootstrap (https://cursos.alura.com.br/course/bootstrap-boas-praticas-no-front-end) disponível na Alura.