12

Opcional: Filtrando elementos com p:selectOneMenu

Nossa lista de livros tende a ser muito grande, uma boa prática seria oferecer para o usuário uma forma de filtrar as informações para que ele veja somente o que lhe interessa. Um exemplo clássico e que é muito comum é quando se vai à uma livraria e nos deparamos com diversas estantes separando os livros pelo gênero. Podemos (e vamos) implementar algo parecido em nossa aplicação. A ideia é que em nossa lista possamos ter um filtro para selecionarmos os livros de um gênero específico, como na imagem abaixo:

Com ajuda do Primefaces implementar esse requisito é algo bem fácil, vamos lá? :)

1) Vamos criar uma nova coluna, para mostrar o gênero na nossa `dataTable` :

```
<p:column headerText="Gênero" sortBy="#{livro.genero}"  filterBy="#{livro.genero}" filterMatchM
    <h:outputText value="#{livro.genero}" />
</p:column>
```

Vamos criar, na classe Livro, um atributo `genero` :

```
// Livro.java
private String genero;

public void setGenero(String genero) {
    this.genero = genero;
}

public String getGenero() {
    return genero;
}
```

2) Dentro dessa coluna, colocaremos o componente `p:selectOneMenu` do primefaces para realizar o filtro:

```
<p:column headerText="Gênero" sortBy="#{livro.genero}" filterBy="#{livro.genero}" filterMatchMo

        <p:selectOneMenu>
            <f:selectItem itemLabel="Selecione..." itemValue="#{null}" noSelectionOption="true"
            <f:selectItems value="#{livroBean.generos}" />
        </p:selectOneMenu>

    <h:outputText value="#{livro.genero}" />
</p:column>
```

O que irá acontecer quando eu selecionar um componente? Por enquanto nada :) Precisamos dizer que queremos filtrar a lista a cada mudança de valor do `p:selectOneMenu`. Para isso, vamos usar o atributo `onchange`:

```
<p:selectOneMenu onchange="PF('tabelaLivros').filter()">
```

Como estamos usando Javascript nesse atributo para controlar o `dataTable`, precisamos identificá-lo com uma variável Javascript usando o `widgetVar`.

No `p:dataTable` coloque o atributo `widgetVar="tabelaLivros"`

```
<p:dataTable value="#{livroBean.livros}" widgetVar="tabelaLivros" var="livro" id="tabelaLivros" paginator="true"
rows="5">
```

4) Para o Primefaces identificar esse componente como um filtro, precisamos passar essa informação através de um `f:facet`

```
<p:column headerText="Gênero" sortBy="#{livro.genero}" filterBy="#{livro.genero}" filterMatchMo
    <f:facet name="filter">
        <p:selectOneMenu onchange="PF('tabelaLivros').filter()">
            <f:selectItem itemLabel="Selecione..." itemValue="#{null}" noSelectionOption="true"
            <f:selectItems value="#{livroBean.generos}" />
         </p:selectOneMenu>
    </f:facet>
    <h:outputText value="#{livro.genero}" />
</p:column>
```

5) Crie uma lista de gêneros no `LivroBean` **(não esqueça de criar o getter)**:

```
// LivroBean.java
private List<String> generos = Arrays.asList("Romance", "Drama", "Ação");

public List<String> getGeneros() {
    return generos;
}
```

6) Coloque o atributo **genero** no modelo `Livro`, não se esqueça dos getters e setters

7) Vamos criar um novo campo no `p:panelGrid` para cadastrarmos agora livros com gênero

```
<p:outputLabel value="Gênero:" for="genero" />
<p:selectOneMenu value="#{livroBean.livro.genero}" id="genero">
    <f:selectItem itemLabel="Selecione..." itemValue="#{null}" />
    <f:selectItems value="#{livroBean.generos}" />
</p:selectOneMenu>
```

8) Reinicie o Tomcat, cadastre um novo livro com algum gênero e faça o teste do filtro :)