09

Opcional: Filtrando outras colunas

Agora vamos filtrar também as colunas isbn, preço e data:

1- Adicione o filtro na coluna do ISBN, o *MatchMode* será `contains`:

```
<p:column headerText="ISBN" sortBy="#{livro.isbn}" filterBy="#{livro.isbn}" filterMatchMode="co
    <h:outputText value="#{livro.isbn}" />
</p:column>
```

2 - Adicione o mesmo filtro na coluna do data:

```
<p:column headerText="Data" sortBy="#{livro.dataLancamento.time}" filterBy="#{livro.dataLancame
    <h:outputText value="#{livro.dataLancamento.time}">
        <f:convertDateTime pattern="dd/MM/yyyy" timeZone="America/Sao_Paulo" />
    </h:outputText>
</p:column>
```

3 - Agora vamos atacar o preço. Adicione o atributo `filterBy="#{livro.preco}"`. Nessa coluna não podemos usar o *matchmode* pois os valores dessa coluna são numéricos. No entanto, podemos plugar um método que decide se um elemento deve ser filtrado ou não! Use o atributo `filterFunction` no componente `p:column` do preço, como listado em baixo:

```
<p:column headerText="Preço" sortBy="#{livro.preco}"  filterBy="#{livro.preco}" filterFunction="
    <h:outputText value="#{livro.preco}">
        <f:convertNumber type="currency" pattern="R$ #0.00" currencySymbol="R$" locale="pt_BR" /
    </h:outputText>
</p:column>
```

4 - Repare que o `filterFunction` faz um binding com um método no `LivroBean`. Crie esse método na classe `LivroBean` com a assinatura seguinte:

```
public boolean precoEhMenor(Object valorColuna, Object filtroDigitado, Locale locale) { //java.

}
```

O primeiro parâmetro é o valor da coluna, o segundo é o filtro, o terceiro define a locale (por exemplo pt, en ou es). Agora precisamos devolver `true` se o `valor` passa pelo `filtro`.

5 - Segue a implementação do método `precoEhMenor` que parece ser complexo mas não é!

```
public boolean precoEhMenor(Object valorColuna, Object filtroDigitado, Locale locale) { // java

        //tirando espaços do filtro
```

```
        String textoDigitado = (filtroDigitado == null) ? null : filtroDigitado.toString().trim

        System.out.println("Filtrando pelo " + textoDigitado + ", Valor do elemento: " + valorC

        // o filtro é nulo ou vazio?
        if (textoDigitado == null || textoDigitado.equals("")) {
            return true;
        }

        // elemento da tabela é nulo?
        if (valorColuna == null) {
            return false;
        }

        try {
            // fazendo o parsing do filtro para converter para Double
            Double precoDigitado = Double.valueOf(textoDigitado);
            Double precoColuna = (Double) valorColuna;

            // comparando os valores, compareTo devolve um valor negativo se o value é menor do
            return precoColuna.compareTo(precoDigitado) < 0;

        } catch (NumberFormatException e) {

            // usuario nao digitou um numero
            return false;
        }
    }
```

Passe pelos comentários para entender o código! Depois tente filtrar pelo preço. Devem aparecer todos os livros que possuem um preço menor do que o filtro.