02

Usando o componente DataList

Transcrição

Com o formulário de cadastro de autores pronto, podemos focar na lista de exibição dos autores.

Assim como no JSF padrão, também há um [`DataTable`][1] no Primefaces, mas no nosso caso, não mostramos muitas informações de cada autor cadastrado na aplicação, apenas seu nome e um link para alterá-lo e removê-lo. Por isso não utilizaremos o `DataTable` , e o Primefaces oferece algo que nos atende melhor, o [`DataList`][2] (vamos deixar o `DataTable` para ser utilizado no `livro.xtml`).

Começaremos com um `DataList` simples, bem parecido com o exemplo do site. Vamos colocá-lo dentro do formulário de exibição dos autores, exibindo os nomes e e-mails:

```
<h:form id="formTabelaAutors">
    <p:dataList value="#{autorBean.autores}" var="autor" type="ordered">
        <f:facet name="header">
            Autores
        </f:facet>

        #{autor.nome} - #{autor.email}
    </p:dataList>
    <!-- h:dataTable -->
</h:form>
```

Mas não queremos números à frente dos autores, isso nós definimos no atributo `type` . Quando `type` é `ordered` , temos os números; `unordered` temos pontos; e o que queremos é `definition` , no qual não aparece nada. Vamos também já colocar os links de alteração e remoção no `dataList` , à frente do nome e e-mail do autor:

```
<h:form id="formTabelaAutors">
    <p:dataList value="#{autorBean.autores}" var="autor" type="definition">
        <f:facet name="header">
            Autores
        </f:facet>

        <h:commandLink value="altera">
            <f:setPropertyActionListener value="#{autor}" target="#{autorBean.autor}" />
        </h:commandLink>

        <h:commandLink value="remove" action="#{autorBean.remover(autor)}" />

        #{autor.nome} - #{autor.email}
    </p:dataList>
    <!-- h:dataTable -->
</h:form>
```

Mas ficou algo bem feio... Pegando inspiração mais uma vez na [página][2] do `DataList` , vemos que podemos utilizar ícones ao invés de texto! Basta remover o atributo `value` e utilizar o atributo `styleClass` , dizendo qual ícone queremos

usar. No exemplo do site, temos `styleClass="ui-icon ui-icon-search"` , e mais algum estilo (que também iremos utilizar) no atributo `style` . Mas esse é o ícone de uma lupa, que não cabe muito na nossa aplicação, como vamos descobrir quais ícones podemos utilizar?

Olhando o código fonte da página, vemos que o Primefaces utiliza por debaixo dos panos o **jQuery**, e é de lá que vêm esses ícones. Então, fazendo uma rápida busca no Google, achamos o [site][3] onde temos todos os ícones disponíveis. Utilizaremos o `ui-icon-pencil` para representar a alteração e o `ui-icon-trash` para representar a remoção do autor:

```
<h:form id="formTabelaAutors">
    <p:dataList value="#{autorBean.autores}" var="autor" type="definition">
        <f:facet name="header">
            Autores
        </f:facet>

        <h:commandLink styleClass="ui-icon ui-icon-pencil" style="float:left;margin-right:10px"
            <f:setPropertyActionListener value="#{autor}" target="#{autorBean.autor}" />
        </h:commandLink>

        <h:commandLink styleClass="ui-icon ui-icon-trash" style="float:left;margin-right:10px"
            action="#{autorBean.remover(autor)}" />

        #{autor.nome} - #{autor.email}
    </p:dataList>
    <!-- h:dataTable -->
</h:form>
```

Ótimo, agora já temos um layout mais apresentável. Mas quando alteramos ou removemos um autor, repara que a página dá uma "piscada", isso porque o nosso `commandLink` não está utilizando ajax, pois não estamos utilizando o `commandLink` do Primefaces. Vamos alterar os `commandLink` s para serem os do Primefaces. Mas lembrando que agora teremos que definir o que queremos atualizar e submeter na página. Nos dois casos queremos submeter somente os dados do próprio link, então utilizaríamos `process="@this"` , mas esse já é o padrão do `commandLink` do Primefaces, então não precisamos explicitar isso, precisamos explicitar apenas o `update` . Na hora de remover, precisamos atualizar o formulário, logo o valor será `@form` e na hora de alterar, temos que atualizar o formulário de cadastro de autores, que tem como id `autor` , logo o valor de `update` será `:autor` :

```
<h:form id="formTabelaAutors">
    <p:dataList value="#{autorBean.autores}" var="autor" type="definition">
        <f:facet name="header">
            Autores
        </f:facet>

        <p:commandLink styleClass="ui-icon ui-icon-pencil" style="float:left;margin-right:10px"
            <f:setPropertyActionListener value="#{autor}" target="#{autorBean.autor}" />
        </p:commandLink>

        <p:commandLink styleClass="ui-icon ui-icon-trash" style="float:left;margin-right:10px"
            action="#{autorBean.remover(autor)}" update="@form" />

        #{autor.nome} - #{autor.email}
    </p:dataList>
    <!-- h:dataTable -->
</h:form>
```

Com o nosso `dataList` pronto, podemos remover a tabela antiga, tornando a tela mais agradável.