

Finalizando os dados do autor

Transcrição

Podemos começar utilizando o `selectOneMenu` do Primefaces, basta trocar o prefixo `h` por `p` e depois alterar o botão de gravar autor, repare que antes estávamos fazendo ajax através do `f:ajax`, mas como o botão do Primefaces já faz ajax por padrão, podemos remover esse componente, mas aproveitando seus atributos, o `execute` e o `render` do `f:ajax` serão o `process` e o `update` do `p:commandButton`, respectivamente. Mas o botão também tem que submeter seus dados, então também adicionaremos `@this` ao `process`:

```
<p:fieldset legend="Dados do Autor">

    <p:outputLabel value="Selecione Autor:" for="autor" />
    <p:selectOneMenu value="#{livroBean.autorId}" id="autor">
        <f:selectItems value="#{livroBean.autores}" var="autor"
            itemLabel="#{autor.nome}" itemValue="#{autor.id}" />
    </p:selectOneMenu>
    <p:commandButton value="Gravar Autor" action="#{livroBean.gravarAutor}"
        process="@this autor" update="tabelaAutores" />
    <!-- restante do código -->
</p:fieldset>
```

Vamos remover o `
`, pois agora queremos que o link para cadastrar novos autores fique ao lado do botão de gravar, além de mudá-lo para usar o Primefaces também. Só que como o link do Primefaces também utiliza ajax, vamos falar que queremos atualizar a página inteira, já que queremos mudar de página, logo o valor de `update` será `@all`:

```
<p:fieldset legend="Dados do Autor">

    <p:outputLabel value="Selecione Autor:" for="autor" />
    <p:selectOneMenu value="#{livroBean.autorId}" id="autor">
        <f:selectItems value="#{livroBean.autores}" var="autor"
            itemLabel="#{autor.nome}" itemValue="#{autor.id}" />
    </p:selectOneMenu>
    <p:commandButton value="Gravar Autor" action="#{livroBean.gravarAutor}"
        process="@this autor" update="tabelaAutores" />

    <p:commandLink value="ou cadastrar novo autor"
        action="#{livroBean.formAutor}" immediate="true" update="@all" />
    <!-- restante do código -->
</p:fieldset>
```

Mas repare que os componentes não estão alinhados, isso porque não temos um container para organizá-los. Então vamos utilizar o `p:panelGrid` para fazer essa organização, como temos 4 componentes, queremos que o `panelGrid` tenha 4 colunas:

```
<p:fieldset legend="Dados do Autor">

    <p:panelGrid columns="4">
```

```

<p:outputLabel value="Selecione Autor:" for="autor" />
<p:selectOneMenu value="#{livroBean.autorId}" id="autor">
    <f:selectItems value="#{livroBean.autores}" var="autor"
        itemLabel="#{autor.nome}" itemValue="#{autor.id}" />
</p:selectOneMenu>
<p:commandButton value="Gravar Autor" action="#{livroBean.gravarAutor}"
    process="@this autor" update="tabelaAutores" />

<p:commandLink value="ou cadastrar novo autor"
    action="#{livroBean.formAutor}" immediate="true" update="@all" />
</p:panelGrid>
<!-- restante do código -->
</p:fieldset>

```

Tabela de autores simples

Agora só falta a tabela que mostra os autores do livro. Nessa tabela iremos fazer somente o mínimo, trocando o prefixo `h` por `p` dos componentes `dataTable` e `column`. E adicionar o atributo `emptyMessage` do `dataTable`, essa mensagem aparecerá quando não houver nenhum dado na tabela, ou seja, nenhum autor cadastrado:

```

<p:dataTable value="#{livroBean.autoresDoLivro}" var="autor"
    id="tabelaAutores" emptyMessage="Nenhum autor">
    <p:column>
        <h:outputText value="#{autor.nome}" />
    </p:column>
    <p:column>
        <h:commandLink value="X" action="#{livroBean.removerAutorDoLivro(autor)}" />
    </p:column>
</p:dataTable>

```

Mas tente remover um autor, repare que ele faz a validação do formulário, isso acontece porque o link de remoção é um `h:commandLink`, e ele submete o formulário inteiro, mas queremos submeter apenas os dados do link, então vamos alterá-lo para ser o `commandLink` do Primefaces e configurar o atributo `process` como `@this`. E o que será atualizado? A tabela toda, então `update` será o id da tabela, `tabelaAutores`:

```

<p:dataTable value="#{livroBean.autoresDoLivro}" var="autor"
    id="tabelaAutores" emptyMessage="Nenhum autor">
    <p:column>
        <h:outputText value="#{autor.nome}" />
    </p:column>
    <p:column>
        <p:commandLink value="X" action="#{livroBean.removerAutorDoLivro(autor)}"
            update="tabelaAutores" process="@this" />
    </p:column>
</p:dataTable>

```

Para finalizar este capítulo, falta atualizar o botão que realmente grava o livro no banco de dados, para ser um componente do Primefaces. Aqui será a mesma coisa que fizemos anteriormente, os atributos `execute` e `render` do `f:ajax` serão o `process` e o `update` do `p:commandButton`:

```
<p:commandButton value="Gravar" action="#{livroBean.gravar}"  
    process="@form" update="@form :formTabelaLivros:tabelaLivros" />
```

Ótimo, o formulário de cadastro de livros está pronto. No próximos capítulo iremos alterar a tabela de exibição dos livros cadastrados, finalizando assim a página `livro.xhtml` .