

Java Server Faces

Tabelas Dinâmicas com o Componente Data Table

Soft Blue
www.softblue.com.br

Todos os direitos de cópia reservados. Não é permitida a distribuição física ou eletrônica deste material sem a permissão expressa e por escrito do autor.

Tópicos Abordados

- A tag *h:dataTable*
- Cabeçalhos, rodapés e títulos
- Estilos das tabelas
- Componentes JSF em tabelas
- Table Models
- A tag *ui:repeat*

- # Tópicos Abordados
-
- A tag *h:dataTable*
 - Cabeçalhos, rodapés e títulos
 - Estilos das tabelas
 - Componentes JSF em tabelas
 - Table Models
 - A tag *ui:repeat*

A Tag `h:dataTable`

- Permite renderizar tabelas utilizando informações dinâmicas

O diagrama ilustra a estrutura da tag `h:dataTable` e como ela se conecta a dados dinâmicos. No topo, um exemplo de tag completa é mostrado: `<h:dataTable value="#{table.contatos}" var="c">`. Uma seta vermelha aponta de `#{table.contatos}` para um retângulo vermelho rotulado "Coleção de itens a serem inseridos". Abaixo, a estrutura interna da tag é detalhada com três exemplos de colunas:

- Coluna 1:** `<h:column>#{c.nome}</h:column>`. Uma seta vermelha aponta de `#{c.nome}` para um retângulo vermelho rotulado "Referência de cada item da coleção".
- Coluna 2:** `<h:column>#{c.email}</h:column>`. Uma seta vermelha aponta de `#{c.email}` para o mesmo retângulo "Referência de cada item da coleção".
- Coluna 3:** `<h:column>#{c.telefone}</h:column>`. Uma seta vermelha aponta de `#{c.telefone}` para um retângulo vermelho rotulado "Conteúdo da coluna".

As setas vermelhas indicam a relação entre as partes da tag e os conceitos de renderização dinâmica: a coleção de dados, a referência ao item atual da coleção e o conteúdo específico a ser exibido em cada coluna.

- ## A Tag `h:dataTable`
- Permite renderizar tabelas utilizando informações dinâmicas
-
- O diagrama ilustra a estrutura da tag `h:dataTable` e como ela se conecta a dados dinâmicos. No topo, um exemplo de tag completa é mostrado: `<h:dataTable value="#{table.contatos}" var="c">`. Uma seta vermelha aponta de `#{table.contatos}` para um texto explicando que é uma coleção de itens a serem inseridos. Abaixo, a estrutura interna da tag é detalhada com três exemplos de colunas:
- Coluna 1:** `<h:column>#{c.nome}</h:column>`. Uma seta vermelha aponta de `#{c.nome}` para um texto explicando que é uma referência de cada item da coleção.
 - Coluna 2:** `<h:column>#{c.email}</h:column>`. Uma seta vermelha aponta de `#{c.email}` para o mesmo texto explicativo.
 - Coluna 3:** `<h:column>#{c.telefone}</h:column>`. Uma seta vermelha aponta de `#{c.telefone}` para um texto explicando que é o conteúdo da coluna.
- Finalmente, uma seta vermelha aponta da tag fechada `</h:dataTable>` para o mesmo texto explicativo sobre o conteúdo da coluna.

A Tag `h:dataTable`

- Permite renderizar tabelas utilizando informações dinâmicas

O diagrama ilustra a estrutura da tag `h:dataTable` e como ela se conecta a dados dinâmicos. No topo, um exemplo de tag completa é mostrado: `<h:dataTable value="#{table.contatos}" var="c">`. Uma seta vermelha aponta de `#{table.contatos}` para um texto explicando que é uma coleção de itens a serem inseridos. Abaixo, a estrutura interna da tag é detalhada com três exemplos de colunas:

- Coluna 1:** `<h:column>#{c.nome}</h:column>`. Uma seta vermelha aponta de `#{c.nome}` para um texto explicando que é uma referência de cada item da coleção.
- Coluna 2:** `<h:column>#{c.email}</h:column>`. Uma seta vermelha aponta de `#{c.email}` para o mesmo texto explicativo.
- Coluna 3:** `<h:column>#{c.telefone}</h:column>`. Uma seta vermelha aponta de `#{c.telefone}` para um texto explicando que é o conteúdo da coluna.

Finalmente, uma seta vermelha aponta da tag fechada `</h:dataTable>` para o mesmo texto explicativo sobre o conteúdo da coluna.

A Tag `h:dataTable`

```
@Named("table")
@SessionScoped
public class TableBean implements Serializable {
    private List<Contato> contatos = new ArrayList<Contato>();

    public TableBean() {
        contatos.add(new Contato(1, "José", "jose@abc.com", "555-3421"));
        contatos.add(new Contato(2, "João", "joao@abc.com", "555-8794"));
    }

    public List<Contato> getContatos() {
        return contatos;
    }
}
```

José	jose@abc.com	555-3421
João	joao@abc.com	555-8794

A Tag `f:facet`

- Permite atribuir informações ao cabeçalho e rodapé das colunas e título da tabela

```
<h:dataTable value="#{table.contatos}" var="c">
  <f:facet name="caption">
    Lista de Contatos
  </f:facet>
  <h:column>
    <f:facet name="header">
      Nome
    </f:facet>
    #{c.nome}
  </h:column>
  ...
</h:dataTable>
```

Os valores podem ser
caption, header ou footer

Lista de Contatos		
Nome	E-mail	Telefone
José	jose@abc.com	555-3421
João	joao@abc.com	555-8794

Estilo das Tabelas

- O estilo das tabelas pode ser definido através de CSS
- A tag `h:dataTable` possui atributos que permitem definir os estilos
 - **styleClass**: para a tabela
 - **headerClass**: para o cabeçalho
 - **footerClass**: para o rodapé
 - **columnClasses**: para cada coluna
 - **rowClasses**: para cada linha

Componentes JSF e Tabelas

- Além de textos, tabelas também podem ter componentes em suas linhas e colunas

Nome	E-mail	Telefone	Excluir
José	jose@abc.com	555-3421	<input type="button" value="Excluir"/>
João	joao@abc.com	555-8794	<input type="button" value="Excluir"/>

```
...  
<h:column>  
  <h:inputText value="#{c.nome}" />  
</h:column>  
...
```

Table Models

- Um table model faz a ponte entre dados a serem exibidos na tabela e a própria tabela



- O próprio JSF encapsula os dados em um table model se o programador não o faz
- Os table model mais utilizados são
 - ArrayDataModel**: usado com arrays de dados
 - ListDataModel**: usado com listas de dados

Benefícios de um Table Model

- Utilizar um table model explicitamente traz algumas facilidades a mais no momento de renderizar tabelas
 - Exibir o número de cada linha renderizada
 - Fazer algum pré-processamento de informações antes que elas sejam exibidas
- Para criar um table model customizado, basta criar uma classe que herda de **DataModel**

A Tag `ui:repeat`

- Alternativa ao uso da tag `h:dataTable`
- Permite renderizar o corpo da tag de forma repetida

```
<table>
  <ui:repeat value="#{table.contatos}" var="c">
    <tr>
      <td>#{c.nome}</td>
      <td>#{c.email}</td>
      <td>#{c.telefone}</td>
    </tr>
  </ui:repeat>
</table>
```

Informações sobre a Iteração

- Com a tag `ui:repeat`, é possível ter acesso a informações relacionadas à iteração

```
<table>
  <ui:repeat value="#{table.contatos}" var="c" varStatus="st">
    <h:panelGroup rendered="#{st.even}">
      <tr>
        <td bgcolor="#FFFFFF">#{st.index + 1}</td>
        <td bgcolor="#FFFFFF">#{c.nome}</td>
      </tr>
    </h:panelGroup>
    <h:panelGroup rendered="#{st.odd}">
      <tr>
        <td bgcolor="#CCCCCC">#{st.index + 1}</td>
        <td bgcolor="#CCCCCC">#{c.nome}</td>
      </tr>
    </h:panelGroup>
  </ui:repeat>
</table>
```

Colocando em Prática...



Agora que você já aprendeu a teoria, acesse as vídeo-aulas práticas e pratique os assuntos abordados neste módulo!

[Clique aqui para acessar as vídeo-aulas práticas](#)
