YourCodeLab.com

Introdução à JSF

Prof. Msc. Alexssander Siqueira

Conteúdo da Aula

- O que é JavaServer Faces?
- Configurações Básicas
- Managed Beans
- Redirecionamento de Páginas

O que é JSF?

- *JavaServer Faces* (JSF) é um framework definido pela especificação JSR-314.

- JSF possui elementos que trabalham com o Controller da aplicação, que são denominados como ManagedBean's.

- Os ManagedBean's substituem os Servlets. Logo, eles são responsáveis por:
  - ✓Obter parâmetros da página
  - ✓Invocar as classes Service
  - ✓Redirecionar o fluxo de tela

Configurações Básicas

- O projeto no Eclipse deve possuir um arquivo web.xml que define o mapeamento do FacesServlet.

- A forma com que as páginas da aplicação são invocadas, também devem ser mapeadas. Logo, uma página JSP com conteúdo pode ser invocada como:
  - ✓/faces/index.jsp
  - ✓*.jsf
  - ✓*.faces
  - ✓*.xhtml

```
<!-- JSF mapping -->
<servlet>
    <servlet-name>Faces Servlet</servlet-name>
    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>

<!-- Map these files with JSF -->
<servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
    <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
```

Configurações Básicas

- O segundo arquivo de configuração é denominado de faces-config.xml que permite determinar a navegação entre as páginas.

```
<navigation-rule>
    <from-view-id>index.jsp</from-view-id>
    <navigation-case>
        <from-outcome>showCustomer</from-outcome>
        <to-view-id>showCustomer.jsp</to-view-id>
        <redirect />
    </navigation-case>
</navigation-rule>
```

Managed Beans

- Um *ManagedBean* é uma classe que funciona como Controller da aplicação.

- A sua definição necessita o uso das tags *@ManagedBean* e *@SessionScoped*. E ainda que a classe seja *Serializable*.

```
@ManagedBean(name = "customerManagedBean")
@SessionScoped
public class CustomerManagedBean implements Serializable{
    private static final long serialVersionUID = -900478543389434700 6L;
```

- Pode-se construir um *ManagedBean* por Caso de Uso.

Managed Beans

- Um *ManagedBean* (MB) pode possuir uma lista de objetos (Model) que podem ser acessados por uma página.

- Outro objeto que o MB pode possuir é uma instancia de uma Model, que pode ser utilizada para obter os atributos de um FORM da página.

Managed Beans

- O construtor da classe MB deve garantir a inicialização das classes a serem manipuladas, evitando a ocorrência de Exceções.

```
@ManagedBean(name = "customerManagedBean")
@SessionScoped
public class CustomerManagedBean implements Serializable{
    private static final long serialVersionUID = -9004785433894347006L;

    private Customer customer;
    private List<Customer> listCustomer;

    private CustomerService service;

    public CustomerManagedBean(){
        service = new CustomerService();
        customer = new Customer(0, "", "");
        listCustomer = service.listAll();
    }
```

Managed Beans

- Um objeto ou atributo de um MB pode ser acessado por uma página de forma semelhante a utilizada em JSTL #{nomeMB.nomeAtributo}.

- A tag @ManagedBean define o nome a ser utilizado pela página para acessar o MB.

- Apenas serão acessos objetos da MB que possuem *get* e *set*.

```
<h:outputLabel value="Name:"/>
<h:inputText value="#{customerManagedBean.customer.name}"/>
```

Managed Beans

- Ao invés de oferecer apenas métodos *doGet* e *doPost*, uma MB pode possuir diversos método que são conhecidos como *Actions*.

- Pode-se criar um método de *Action* para cada ação de negócio ou método de CRUD.

```
public void searchByNameAction(){
    System.out.println("Searching...");
    this.listCustomer = service.findCustomerByName(customer.getName());
}

public String insertCustomerAction(){
    System.out.println("Saving...");
    service.insertCustomer(customer);

    return "index";
}
```

Redirecionamento de Páginas

- Um método Action pode retornar void, neste caso após sua execução, o fluxo retornará sempre para a mesma página.

- Quando a Action é implementada com retorno String, é possível mapear uma regra de navegação no arquivo faces-config.

```
<navigation-rule>
    <from-view-id>newCustomer.jsp</from-view-id>
    <navigation-case>
        <from-outcome>index</from-outcome>
        <to-view-id>index.jsp</to-view-id>
        <redirect />
    </navigation-case>
</navigation-rule>
```

Redirecionamento de Páginas

- Caso o desenvolvedor queira utilizar um botão para redirecionar para um destino mapeado, pode-se usar a chamada do botão Cancel.

```
<h:commandLink styleClass="btn btn-primary" value="Save" action="#{customerManagedBean.updateCustomerAction}"/>
<h:commandLink styleClass="btn btn-default" value="Cancel" action="index?faces-redirect=true"/>
```

- Repare que o botão Save chama um método *Action* da classe MB.

YourCodeLab.com

Questions???

alexssander_as@hotmail.com