JSF 2.0 na Prática – Parte II

14/08/2010 Diego NunesDeixe um comentárioIr para os comentários

Boa noite pessoal.
Neste post iremos dar continuidade a nossa série de posts sobre JSF 2.0. Irei apresentar aqui novos recursos do JSF 2.0, como declaração de Managed Beans, navegação e acesso a recursos.

Obs.: Iremos utilizar o nosso projeto criado no primeiro post da série, JSF 2.0 na Prática – Parte I.

Anotações para declaração de Managed Beans
Nas versão 2.0 do JSF não precisamos mais realizar declarações de ManagedBean via XML no arquivo faces-config.xml, podemos agora realizar este registro via anotação. Veremos logo abaixo estes recursos:

@ManagedBean > Esta anotação é utilizada para podermos indicar que a classe é um Manged Bean, nela temos duas propriedades, ***name*** que indica o nome do nosso Managed Bean (quando não informado essa propriedade o nosso Managed Bean recebe o nome da nossa classe com o primeiro caracter minusculo) e a propriedade ***eager*** que quando o seu valor for **true** e o scopo do nosso ManagedBean for de **Aplicação** indica que este bean deve ser inicializado quando a aplicação for iniciada.

Escopos do ManagedBean:

Os escopos dos nossos Managed Beans também iremos declara-los via anotações, se no nosso Managed Bean não colocarmos nenhuma declaração de escopo é atribuido por default o **scopo de requisição** (ResquestScoped). Abaixo os escopos permitidos para os nossos Managed Beans:

@ApplicationScoped > Indica que o escopo do nosso Managed Bean será de aplicação.

@SessionScoped > Indica que o escopo do nosso Managed Bean será de sessão.

@RequestScoped > Indica que o escopo do nosso Managed Bean será de requisição.

@ViewScoped > Indica que o escopo do nosso Managed Bean será de visão.

@NoneScoped > Indica que o escopo do nosso Managed Bean não terá escopo.

@CustomScoped > Indica que o escopo do nosso Managed Bean terá um escopo personalizado, criado pelo desenvolvedor.

Exemplo de Declaração de ManagedBean

JSF 1 via faces-config.xml

<managed-bean>
< managed-bean-class>br.com.ClienteBean</managed-bean-class>
< managed-bean-name>clienteBean</managed-bean-name>
< managed-bean-scope>request</managed-bean-scope>
< /managed-bean>

JSF 2 via anotação

@ManagedBean(name=”clienteBean”)
@RequestScoped
public class ClienteBean {
}

Navegação

No JSF 2 a navegação entre páginas foi simplificada, temos agora um recurso de navegação
implicita, não sendo necessário mais configurações de navegação no arquivo **facesconfig.**
xml, abaixo segue uma comparação:

JSF 1

<navigation-rule>
< navigation-case>
< from-view-id>cadastroCliente.xhtml</from-view-id>
< outcome>sucesso</outcome>
< to-view-id>cadastroClienteSucesso.xhtml</to-view-id>
< /navigation-case>
< /navigation-rule>

JSF 2

```
public String incluirCliente(){
// Regras Aqui
return “cadastroClienteSucesso”;
}
ou
<h:commandButton id=”redirectCadastro”  action=”cadastroClienteSucesso”/>
```

Acesso a recursos

No JSF 2.0 temos padrões de acesso a recursos da nossa aplicação, como imagens, arquivos.js, etc. A apenas um requisito, estes arquivos devem estar em um diretório **resources** da nossa aplicação.
Para acesso a recurso de estilos (arquivos.css) temos a tag **<h:outputStylesheet/>** e para acesso a recursos de script (arquivos .js) temos a tag **<h:outputScript/>.**

Nestas duas tags temos três atributos importantes:

library > Indica o diretório onde o recurso se encontra, por exemplo, se temos o seguinte diretório resources/css/estilos.css teremos **library=”css”**.

name > Indica o nome do atributo em si, por exemplo, **ame=”estilos.css”.**

target > Um recurso interessante, podemos indicar através do atributo target onde nosso recurso será alocado, por exemplo, se colocarmos **target=”body”** este recurso será alocado no corpo de nossa página e sempre que ela for carregada ele será executado agora se colocarmos **target=”head”** ele somente será executado quando chamado.

Exemplo:

```
<h:body>
< h:outputStylesheet library=”css” name=”estilos.css” target=”body”/>
<h:outputScript library=”javascript” name=”scripts.js” target=”head”/>
…
< /h:body>
```

Também podemos acessar recursos através de EL, a sintax para este acesso é **resource['LIBRARY:NAME']**, LIBRARY e NAME correspondem aos atributos de mesmo nome das tags **<h:outputStylesheet/> e <h:outputScript/>.**

Exemplo:
<h:graphicImage value="#{**resource['images:imagem.gif']**}"/>

ou

<h:graphicImage name="images/imagem.gif"/>

Mão na Massa

Vamos dar continuidade ao nosso projeto criado na parte 1 desta séria inserindo no mesmo os novos recursos apresentados aqui.

Criando o Managed Bean

No nosso projeto clique em **New > Class**, em **name** digite ClienteBean e em **Packege** digite br.com.jsf.managedbean

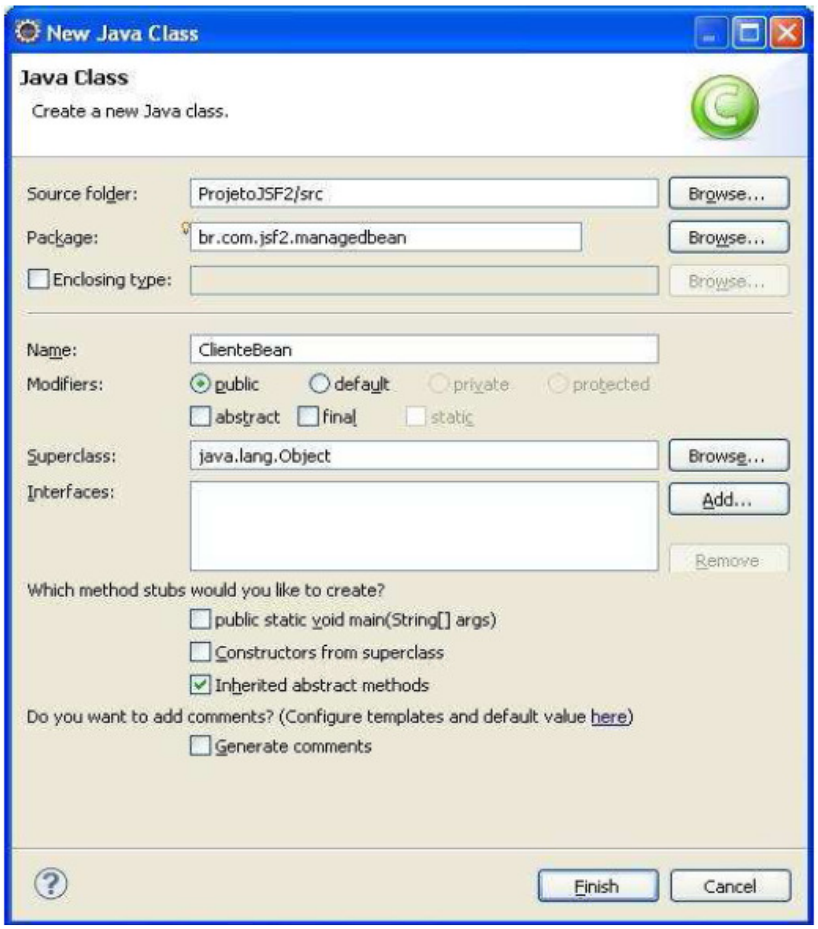


Imagem 1 - Criando nosso Managed Bean
A nossa classe deverá ficar conforme a seguir:

```
package br.com.jsf2.managedbean;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;

@ManagedBean(name=”clienteBean”)
@RequestScoped
public class ClienteBean {

private String nome;
private String idade;

public String cadastrarCliente(){
return “clienteCadastrado”;
}

public String getNome() {
return nome;
}

public void setNome(String nome) {
this.nome = nome;
}

public String getIdade() {
return idade;
}

public void setIdade(String idade) {
this.idade = idade;
}

}
```

No código acima criamos o nosso Managed Bean com dois atributos, **nome** e **idade** e criamos também um método com o nome **cadastrarCliente** (o retorno do método nos redirecionará para a página **clienteCadastrado.xhtml**) .
Agora vamos criar uma nova página para nossa aplicação. Clique com o botão direito em nosso projeto e selecione **New > HTML File**. Em **File name** digite cadastroCliente.xhtml e clique **Finish**.

Altere a nossa página criada colocando o código a seguir:

```
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0
Transitional//EN” “http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd”>
<html xmlns=”http://www.w3.org/1999/xhtml”
xmlns:h=”http://java.sun.com/jsf/html”>
<h:head><title>Cadastro de Cliente</title></h:head>
< h:body>
<h:form id=”formCadastroCliente”>
Nome: <h:inputText id=”nome” value=”#{clienteBean.nome}”/><br/>
Idade: <h:inputText id=”idade”
value=”#{clienteBean.idade}”/><br/><br/>
<h:commandButton id=”cadastrarCliente”
action=”#{clienteBean.cadastrarCliente}”
image=”#{resource['imagens:gravar.png']}”/>
< /h:form>
< /h:body>
< /html>
```

Criamos a nossa página de cadastro com os nossos dois atributos e colocamos também um botão que executará o método **cadastrarCliente** do nosso Managed Bean.
Vamos agora criar a nossa página no qual vamos redirecionar quando o cliente for cadastrado. Clique com o botão direito em nosso projeto e selecione **New > HTML File**. Em **File name** digite lienteCadastrado.xhtml e clique **Finish**.

Altere o código da página criada conforme abaixo:

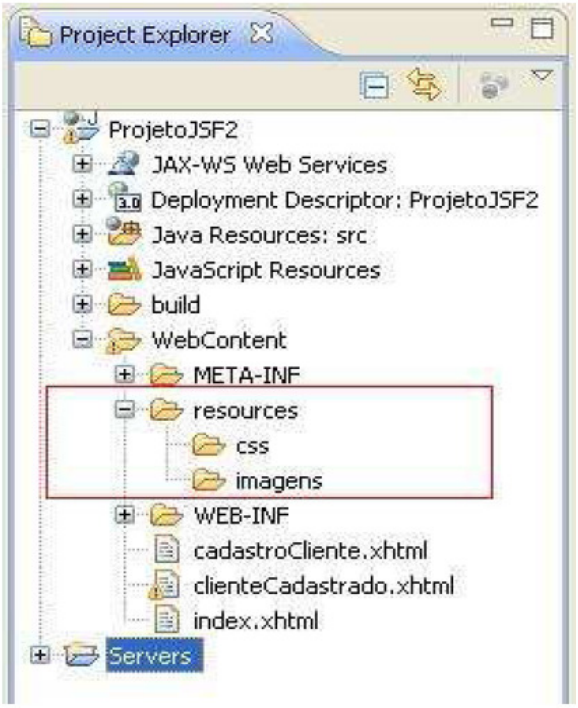
```
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0
Transitional//EN”
“http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”>
<html xmlns=”http://www.w3.org/1999/xhtml”
xmlns:h=”http://java.sun.com/jsf/html”>
<h:head><title>JSF 2.0</title></h:head>
< h:body>
< h:outputStylesheet library=”css” name=”estilos.css” target=”body”/>
<div>
Cliente Cadastrado com Sucesso!!!
< /div>
<h:form>
```

< h:commandButton id=”voltar” value=”Cadastrar Cliente”
action=”cadastroCliente”/>
< /h:form>
</h:body>
< /html>

Nesta página será exibida quando na tela de cadastro for clicado o botão Incluir, nesta página utilizamos a tag **h:outputStylesheet** para importação de um arquivo de estilos (criaremos a seguir este arquivo) e também criamos um botão para que seja retornado a página de cadastro, repare que não vamos executar um método neste botão, na propriedade action colocamos **diretamente o nome da nossa página** a ser redirecionada **“cadastroCliente”**.

Vamos criar agora uma pasta para colocarmos os recursos da nossa aplicação (imagens e css). Clique com o botão direito na pasta **WebContent** do nosso projeto e selecione **New > Folder** em **Folder Name** digite resource, na pasta criada crie duas pastas dentro da mesma,
com os nomes **imagens** e **css**, deverá ficar como a seguir:

Imagem 2 - Pasta de recursos da aplicação



Vamos criar um arquivo de estilos na pasta **css** que será utilizado na página clienteCadastrado.xhtml, clique com o botão direito na mesma e selecione **New > Other > Web > CSS File**, na janela que aparecerá em **File name** digite estilos e clique em **Finish**.

Altere o código do nosso arquivo recém criado para:

```
div {
padding-bottom: 10px;
text-align: left;
font-weight: bold;
color: blue;
}
```

Vamos colocar na nossa pasta imagens um arquivo .png com o nome **gravar.jpg**, esta imagem no nosso botão de gravação da nossa página cadastrarCliente.xhtml. Para rodar o nosso exemplo clique com o botão direito em nossa página cadastrarCliente.xhtml e selecione **Run As>Run on Server,** selecione o nosso servidor Apache Tomcat 7.0 e clique em **Finish.**

O resultado é apresentado logo a seguir:

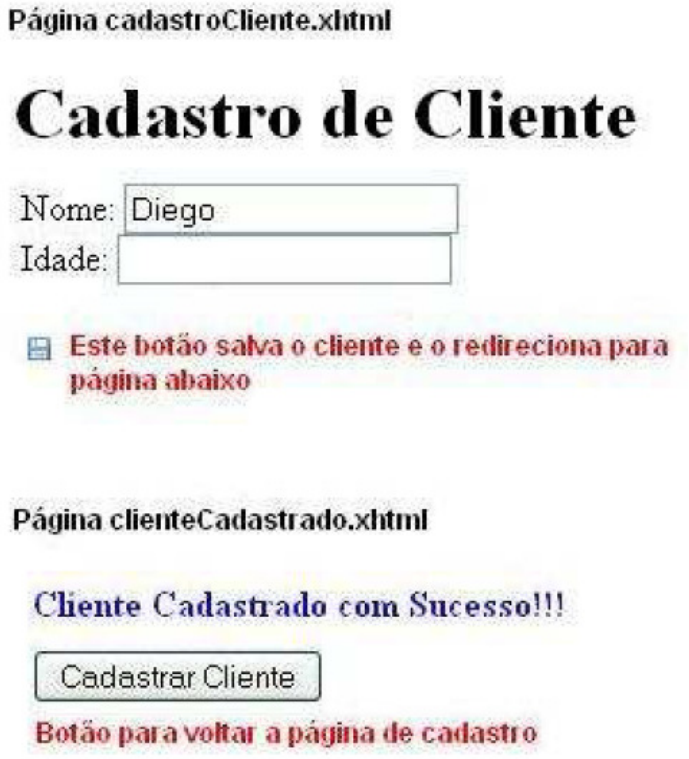


Imagem 3 – Resultado

Bom pessoal, espero que vocês possar tirar proveito dos recursos aprsentados aqui, em breve postarei o terceiro post desta série para apresentar mais recursos do JSF 2.0.
Até.