JSTL -

Tags de Finalidades Gerais: Iterator

Iteração com coleções de objetos é outra ocorrência muito comum em páginas JSP

Ao invés de criar scriptlets podemos utilizar tags JSTL, que simplifica bastante o desenvolvimento da página JSP

<c:forEach>

Finalidade

Repete o conteúdo baseado em uma coleção de objetos ou repete o conteúdo um certo número fixo de vezes

O conteúdo do corpo das tags pode ser qualquer código JSP válido!

Sintaxe

A sintaxe da ação `<c:forEach>` é:

Sintaxe 1: Iteração baseada em uma coleção de objetos

```
<c:forEach[var="varName"] items="collection"
[varStatus="varStatusName"]
[begin="begin"] [end="end"] [step="step"]>
     body content
</c:forEach>
```

Sintaxe 2: Iteração baseada em um numero fixo de vezes

```
<c:forEach [var="varName"] [varStatus="varStatusName"]
begin="begin" end="end" [step="step"]>
body content
</c:forEach>
```

Atributos

Nome	Tipo	Descrição
Var	String	Nome da variavel de escopo.
Items	Qualquer tipo suportado. Vide tópico seguinte	Coleção de itens à ser iteragido.
varStatus	String	Nome da variavel, que mantem o status da operação
Begin	int	Valor inicial da iteração
End	int	Valor final da iteração

Step	int	Passo em que ocorrerá o incremento

Tipos de Coleções suportados

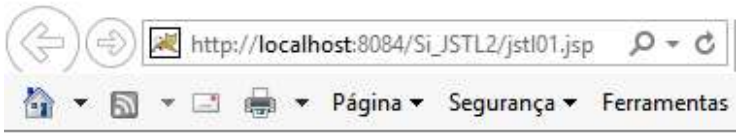
- Arrays
- Implementações do java.util.Collection.
- Implementações do java.util.Iterator.
- Implementações do java.util.Enumeration.
- Implementações do java.util.Map
- String

Exemplo forEach

jstl01.jsp

```jsp
<%@page contentType="text/html" pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c"  uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
        <title>Utilizando JSTL - forEach</title>
        <link  href="estilo.css" type="text/css" rel="stylesheet">
    </head>
    <body>
        <h2>Lista de Cursos</h2>
        <c:set var="cursos" value="Internet, Redes, Engenharia, Técnico" scope="page"/>
        <c:forEach var="curso" items="${cursos}">
            <c:out value="${curso}"/><br>
        </c:forEach>
        <hr>
        <c:forEach var="i" begin="1" end="5" >
            <c:out value="${i}"/><br>
        </c:forEach>
        <hr>
    </body>
</html>
```

Resultado



Lista de Cursos

Internet
Redes
Engenharia
Técnico

1
2
3
4
5

Exemplo forEach com Bean

Cursos.java

```java
package br.ifmt.dai.curso;
public class Cursos {
    public String[] getListaCursos(){
        String[] lista={"Internet","Redes","Engenharia","Técnico"};
        return lista;
    }
}
```

jstl02.jsp

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Utilizando JSTL - forEach</title>
        <link  href="estilo.css" type="text/css" rel="stylesheet">
    </head>
    <body>
        <h2>Lista de Cursos</h2>
        <jsp:useBean id="curso" class="br.ifmt.dai.curso.Cursos" />
        <c:set var="cursos" value="${curso.listaCursos}" scope="session" />
        <c:forEach var="nome" items="${cursos}">
            <c:out value="${nome}<br />" escapeXml="false" />
        </c:forEach>
    </body>
</html>
```

Resultado

Tags de Finalidades Gerais: forTokens

<c: forTokens >

Finalidade

Interagir sobre palavras separadas por um certo delimitador

Sintaxe

A sintaxe da ação **<c:forTokens>** é:

Sintaxe

```
<c:forTokens items="stringOfTokens" delims="delimiters"
[var="varName"]
[varStatus="varStatusName"]
[begin="begin"] [end="end"] [step="step"]>
body content
</c:forTokens>
```

Atributos

Nome	Tipo	Descrição
Var	String	Nome da variavel de escopo.
Items	Qualquer tipo suportado. Vide tópico seguinte	Coleção de itens à ser iteragido.
Delims	String	Delimitador entre as palavras
varStatus	String	Nome da variavel, que mantem o status da operação
Begin	int	Valor inicial da iteração
End	int	Valor final da iteração
Step	int	Passo em que ocorrerá o incremento

Exemplo forTokens

jstl03.jsp

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Utilizando JSTL - forTokens</title>
        <link  href="estilo.css" type="text/css" rel="stylesheet">
    </head>
    <body>
        <h2>Lista de Cursos</h2>
        <c:set var="cursos" value="Internet, Redes, Engenharia, Técnico" scope="page" />
        <c:forTokens var="curso" delims="," items="${cursos}">
            <c:out value="${curso}"/><br>
        </c:forTokens>
    </body>
</html>
```

Resultado

Tags relacionadas à URL

Tags relacionadas à URL
Links , importar e redirecionar são funcionalidades necessárias as páginas JSP

O JSTL dispõem de um conjunto de tags, bem faceis de usar, para simplificar estas tarefas

`<c:import>`
Finalidade
Importar o conteúdo de uma página para a página atual

Sintaxe
A sintaxe da ação `<c:import>` é:

Sintaxe 1

```
<c:import url="url" [context="context"]
[var="varName"] [scope="{page|request|session|application}"]
[charEncoding="charEncoding"]>
optional body content for <c:param> subtags
</c:import>
```

Sintaxe 2

```
<c:import url="url" [context="context"]
varReader="varReaderName"
[charEncoding="charEncoding"]>
body content where varReader is consumed by another action
</c:import>
```

Atributos

Nome	Tipo	Descrição
url	String	A URL contendo os recursos à serem importados

context	String	Nome do contexto, quando acessando uma url que pertence à outro contexto
var	String	Nome da variavel de escopo
scope	String	Escopo da variavel
charEncoding	String	Código do caracter
varReader	String	Nome da variavel

Exemplo import

titulocursos.html

```
<html>
    <head>
        <title>Utilizando JSTL - import</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <table>
            <tr>
                <td><a href="Internet.jsp" >Internet</a></td>
                <td><a href="Redes.jsp" >Redes</a></td>
                <td><a href="EngComp.jsp" >Engenharia</a></td>
            </tr>

        </table>
    </body>
</html>
```

jstl04.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Utilizando JSTL - import</title>
        <link  href="estilo.css" type="text/css" rel="stylesheet">
    </head>
    <body>
        <h2>Lista de cursos</h2>
        <c:import url="/titulocursos.html"/>
    </body>
</html>
```

Resultado

Tags de Finalidades Gerais: url

`<c: url >`
Finalidade
Criar uma URL

Sintaxe
A sintaxe da ação `<c:url>` é:

Sintaxe 1 sem corpo

```
<c:url value="value" [context="context"]
[var="varName"] [scope="{page|request|session|application}"]/>
```

Sintaxe 2 com corpo

```
<c:url value="value" [context="context"]
[var="varName"] [scope="{page|request|session|application}"]>
<c:param> subtags
</c:url>
```

Atributos

Nome	Tipo	Descrição
value	String	URL à ser processada.
context	String	Nome do contexto, quando acessando uma url que pertence à outro contexto
var	String	Nome da variavel de escopo
scope	String	Escopo da variavel

Exemplo url

jstl05.jsp

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Utilizando JSTL - url</title>
        <link  href="estilo.css" type="text/css" rel="stylesheet">
    </head>
    <body>
        <h2>Cadastrar Alunos</h2>
        <c:url value="cad2.jsp" var="url">
            <c:param name="nome" value="Aluno"/>
            <c:param name="empresa" value="Dai"/>
        </c:url>
        <a href="${url}">Incluir</a>
        <hr>
        Valor da url: <c:out value="${url}" />
    </body>
</html>
```

Resultado

Tags de Finalidades Gerais: redirect

`<c: redirect >`

Finalidade

Redirecionar para outra URL

Sintaxe

A sintaxe da ação `<c:redirect>` é:

Sintaxe 1 sem corpo

```
<c:redirect url="value" [context="context"]/>
```

Sintaxe 2 com corpo

```
<c:redirect url="value" [context="context"]/>
<c:param> subtags
</c:redirect>
```

Atributos

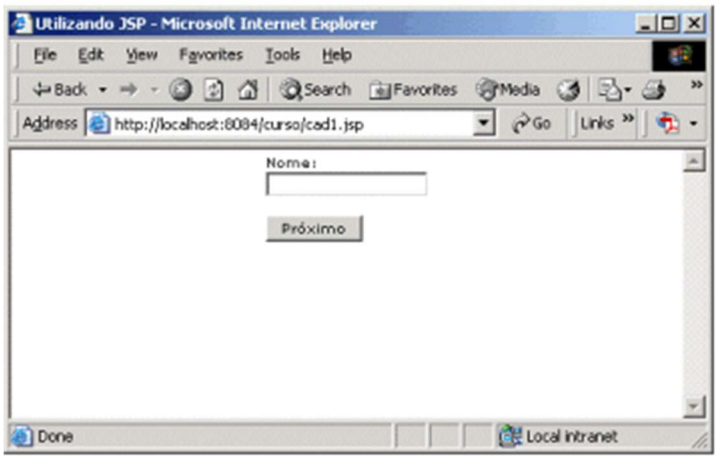
Nome	Tipo	Descrição
url	String	A URL a ser redirecionada
context	String	Nome do contexto, quando acessando uma url que pertence à outro contexto

Exemplo redirect

jstl06.jsp

```
1  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
2  <html>
3      <head>
4          <title>Utilizando JSTL - redirect</title>
5          <LINK href="estilo.css" type="text/css" rel="stylesheet">
6      </head>
7      <body>
8          <h2>Cadastrar Aluno</h2>
9
10         <c:redirect url="cadl.jsp"/>
11
12     </body>
13 </html>
```

Resultado



Tags de Finalidades Gerais: param

```
<c: param >
```

Finalidade
Adicionar parametros para as tags `<c:import>`, `<c:url>` e `<c:redirect>`

Sintaxe
A sintaxe da ação <c:param> é:

Sintaxe 1 sem corpo

```
<c:param name="name" value="value"/>
```

Sintaxe 2 com corpo

```
<c:param name="name">
parameter value
</c:param>
```

Atributos

Nome	Tipo	Descrição
name	String	Nome do parametro.
value	String	Valor do parametro.

Exemplo param

jstl07.jsp

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Utilizando JSTL - redirect</title>
        <link  href="estilo.css" type="text/css" rel="stylesheet">
    </head>
    <body>
        <h2>Cadastrar Alunos</h2>
        <c:redirect url="cad3.jsp">
            <c:param name="nome" value="Aluno"/>
            <c:param name="email" value="aluno@ifmt.br"/>
        </c:redirect>
    </body>
</html>
```

Resultado

Tags para Internacionalização - Formatação de Números

Formatando Números

A tag <fmt:formatNumber> permite formatar numeros, moedas e percentuais de acordo com a cultura do browser

Por exemplo

```jsp
<fmt:formatNumber value="543.21" type="currency"/><br>
```

Se a cultura do browser estiver configurad para pt-BR o resultado será:

R$ 543,21

Se a cultura do browser estiver configurad para en-US o resultado será:

$ 543.21

Tags para Internacionalização - Formatação de Datas

Formatando Datas

A tag <fmt:formatDate> permite formatar datas e horários com a cultura do browser

Por exemplo

```jsp
<fmt:formatDate value="${date}" timeStyle="long" dateStyle="long"/><br>
```

Se a cultura do browser estiver configurad para pt-BR o resultado será:

18 de Setembro de 2006

Se a cultura do browser estiver configurad para en-US o resultado será:

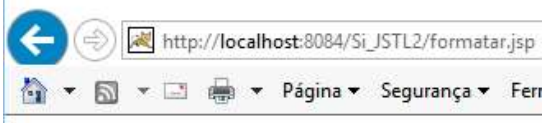
June 23, 2017

Exemplo de formatação

formatar.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<jsp:useBean id="data" class="java.util.Date" />
<html>
    <body>
        <hr>Padrão do Navegador<br>
        <fmt:formatNumber value="543.21" type="currency"/><br>
        <fmt:formatNumber value="12.3" pattern=".000"/><br>
        <fmt:formatNumber value="123456.7891" pattern="#,#00.0#"/><br>
        <fmt:formatDate value="${data}" timeStyle="long" dateStyle="long" /><br>
        <fmt:formatDate value="${data}" pattern="dd.MM.yy" />
        <hr>Ingles<br>
        <fmt:setLocale value="en_US"/>
        <fmt:formatNumber value="543.21" type="currency"/><br>
        <fmt:formatNumber value="12.3" pattern=".000"/><br>
        <fmt:formatNumber value="123456.7891" pattern="#,#00.0#"/><br>
        <fmt:formatDate value="${data}" timeStyle="long" dateStyle="long" /><br>
        <fmt:formatDate value="${data}" pattern="MM.dd.yy" />
        <hr>Frances<br>
        <fmt:setLocale value="fr_FR"/>
        <fmt:formatNumber value="543.21" type="currency"/><br>
        <fmt:formatNumber value="12.3" pattern=".000"/><br>
        <fmt:formatNumber value="123456.7891" pattern="#,#00.0#"/><br>
        <fmt:formatDate value="${data}" timeStyle="long" dateStyle="long" /><br>
        <fmt:formatDate value="${data}" pattern="dd.MM.yy" />
    </body>
```

Resultado



Padrão do Navegador
R$ 543,21
12,300
123.456,79
23 de Junho de 2017
23.06.17

Ingles
$543.21
12.300
123,456.79
June 23, 2017
06.23.17

Frances
543,21 €
12,300
123 456,79
23 juin 2017
23.06.17

Tags para ações SQL

Para ações com banco de dados é recomendável utilizar classes para acesso à banco de dados, de preferência utilizando a arquitetura MVC, mas pode haver a necessidade de utilizar diretamente instruções SQL diretamente na página JSP (pequenas aplicações, protótipos, etc).

As tags JSTL disponibilizam algumas funcionalidades básicas para interagir com banco de dados.

Para utilizarmos estas tags é necessário adicionar a seguinte referencia:

```
<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql" %>
```

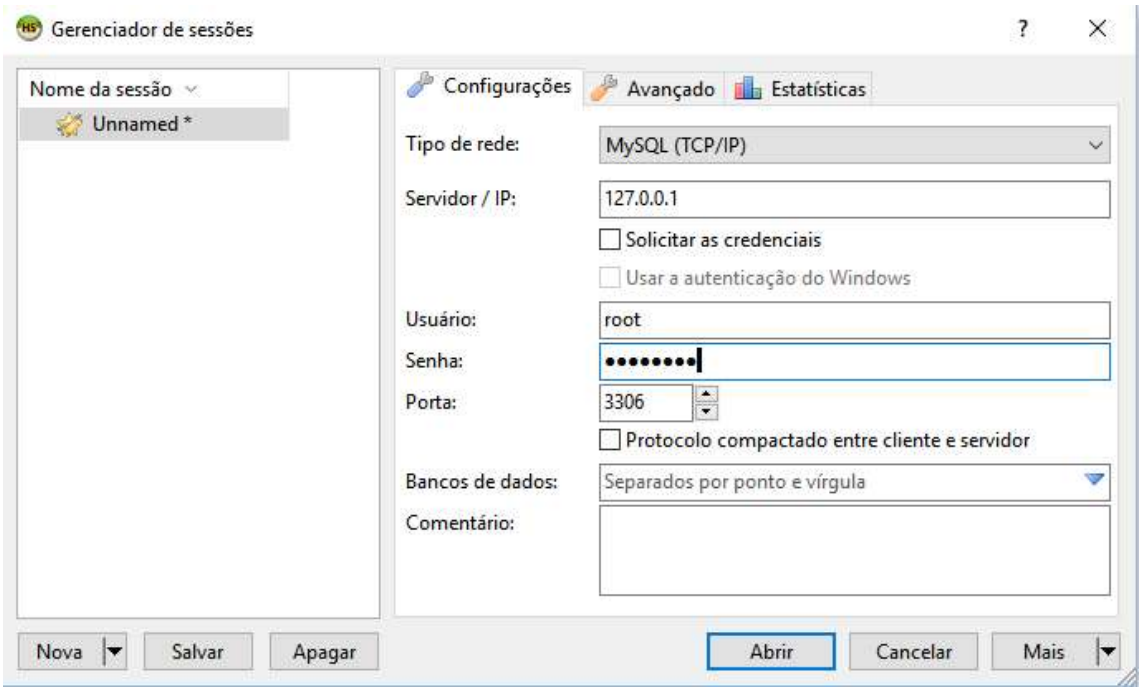
Tags para ações SQL - Data Source

As operações sobre o banco de dados são executadas através de um objeto do tipo Data Source.
Este objeto é responsável pela conexão com o banco de dados.

Para criar o Data Source

```
<sql:setDataSource
    var="conexao"
    dataSource="jdbc:mysql://127.0.0.1/loja,com.mysql.jdbc.Driver,root,123"
/>
```

A informações passadas para o atributo dataSource são as mesmas utilizadas na sua conexão via HeidiSQL



Tags para ações SQL - Consultando Dados

No caso vamos utilizar a tabela categorias para fazer uma consulta.

No query browser ficaria da seguinte forma:

A tag <sql:query> permite realizar consultas no banco de dados.

Observe no código abaixo que criamos a instrução SQL utilizando o dataSource criado anteriormente
na variável conexao.

```
<sql:query dataSource="${conexao}" var="consulta">
    select idcategoria, categoria from categorias order by categoria
</sql:query>
```

Para apresentar os dados na tela basta obter o valor das colunas a partir da variável consulta.

```
<table>
    <c:forEach var="linha" items="${consulta.rows}">
        <tr>
            <td><c:out value="${linha.idcategoria}"/></td>
            <td><c:out value="${linha.categoria}"/></td>
        </tr>
    </c:forEach>
</table>
```

e à partir de cada linha informar o nome da coluna que você quer ler.

Tags para ações SQL - Atualizando os Dados

A tag <sql:update> permite realizar atualizações no banco de dados

```
<c:if test='${!empty param.categoria}'>
    <sql:update dataSource="${conexao}" var="alteracao">
        insert categorias (categoria) values(?)
        <sql:param value="${param.categoria}"/>
    </sql:update>
    <c:out value="${'Incluido com sucesso!'}" />
</c:if>
```

Observe que utilizamos parâmetros nesta operação, onde cada interrogação na instrução SQL é um parâmetro, que deve ser atribuído algum valor através da tag <sql:param>

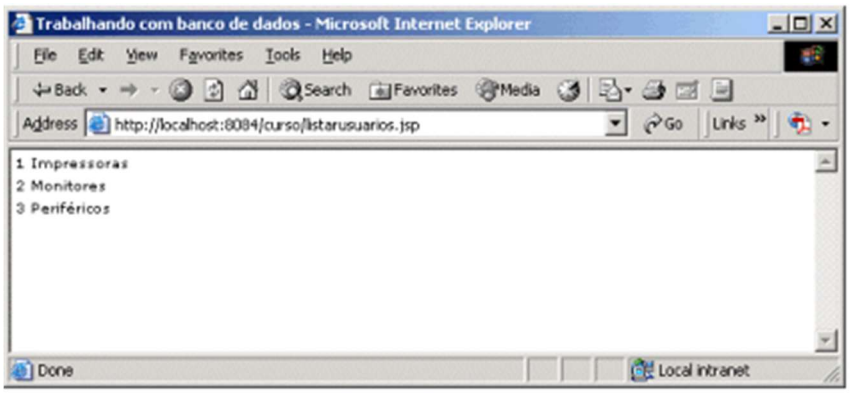
Exemplo completo de Banco de Dados

listarcategorias.jsp

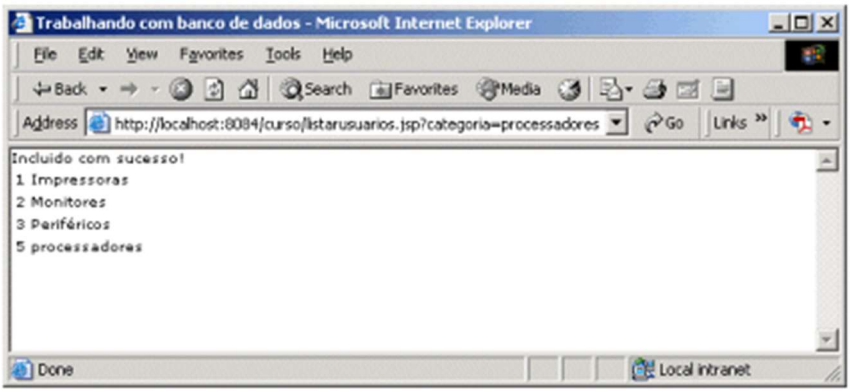
```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql" %>
<html>
    <body>
        <c:if test="${idioma==null}">
            <fmt:setBundle basename="br.com.learning.curso.Recursos"
                           var="idioma" scope="page"/>
        </c:if>
        <sql:setDataSource var="conexao" driver="com.mysql.jdbc.Driver"
            url="jdbc:mysql://127.0.0.1:3306/loja"
            user="root" password="ifmt2k17"/>
        <sql:query dataSource="${conexao}" var="consulta">
            select idcategoria, categoria from categorias
        </sql:query>
            <table border="0" align="center">
                <tr><td colspan="2" align="center"><fmt:message key="login.apresentacao"
                          bundle="${idioma}"/><font size="12"></td></tr>
                <tr><td colspan="2" align="center"><a href="incluircategorias.jsp">
                          Incluir nova categoria</td></tr>
                <c:forEach var="linha" items="${consulta.rows}">
                    <tr>
                        <c:url value="mantercategorias.jsp" var="url1">
                            <c:param name="idcategoria" value="${linha.idcategoria}"/>
                            <c:param name="categoria" value="${linha.categoria}"/>
                        </c:url>
                        <c:url value="excluircategorias.jsp" var="url2">
                            <c:param name="idcategoria" value="${linha.idcategoria}"/>
                            <c:param name="categoria" value="${linha.categoria}"/>
                        </c:url>
                        <td><a href="${url1}"><c:out value="${linha.categoria}"/></td>
                        <td><a href="${url2}">Excluir</td>
                    </tr>
                </c:forEach>
            </table>
    </body>
</html>
```

Resultado do exemplo de Banco de Dados

Sem parâmetro



Com parâmetro



Avaliação da Aula

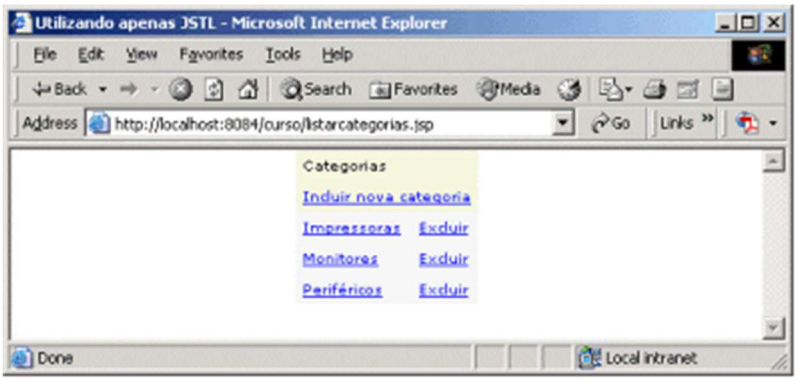
Neste exercicio utilizar apenas JSTL

Não utilizar scriptles (<% %>)

1 - Criar a página listarcategorias.jsp

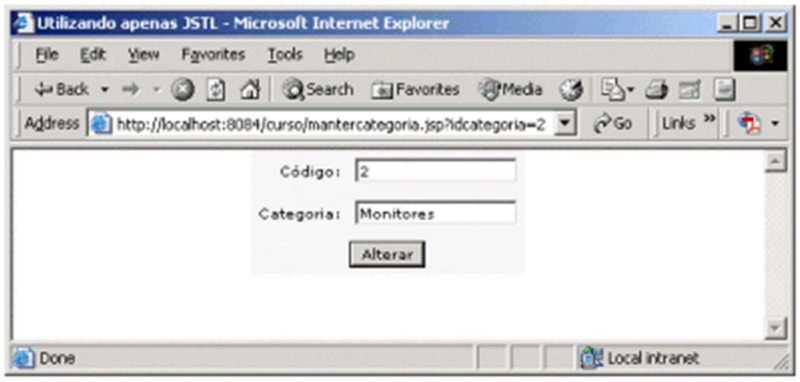
Esta página apresenta links para as seguintes páginas:

- mantercategorias.jsp
- incluircategorias.jsp
- excluircategorias.jsp



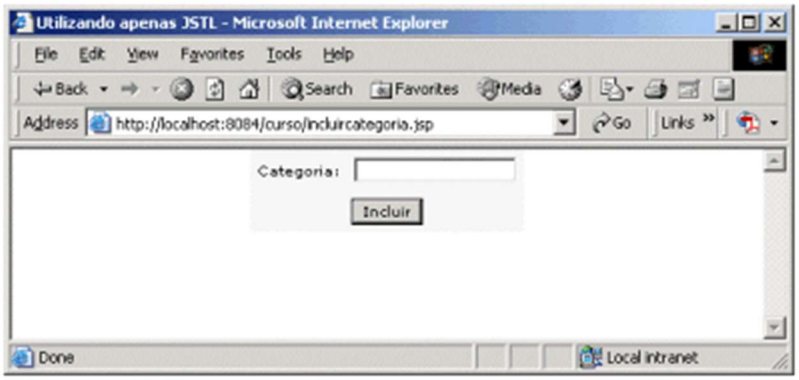
2 - Criar a página mantercategorias.jsp

Altera o texto da categoria
Redireciona automaticamente para listarcategorias.jsp



3 - Criar a página incluircategorias.jsp

Incluir uma nova categoria
Redireciona automaticamente para listarcategorias.jsp



4 - Criar a página excluircategorias.jsp

Excluir a categoria
Redireciona automaticamente para listarcategorias.jsp