MC0037 – Programação para Web

Aula 5: JSP – Java Server Pages

2 Q / 2018

Introdução

Aulas passadas

- POO
- Banco de Dados
- Hibernate
- MVC – SpringMVC
- Automatização e configuração – SpringBoot
- HTML

- Não focamos no dinamização dos dados...apresentação dos dados...interface

Aula de hoje: Roteiro

- JSP – Java Server Pages

JSP – JavaServer Pages

- Um JSP é uma página HTML com algum conteúdo dinâmico em Java que é embutido na mesma utilizando algumas tags especiais
-
- Essas tags especiais são compiladas por um compilador JSP (JSP engine) do servidor de aplicações
-
- Páginas JSP são convertidas em servlets pelo JSP engine

Escrevendo uma página JSP

- Exemplo: página JSP para apresentar a data e hora:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Primeiro JSP</title>
</head>
<body>
    <h2>Bem-vindo à Primeira Página JSP!</h2>
    <p>Data e hora atuais:
    <%= new java.util.Date() %></p>
</body>
</html>
```

Expressão JSP, adciona conteúdo dinâmico à pagina

Escrevendo uma página JSP

- Exemplo: página JSP para apresentar a data e hora:

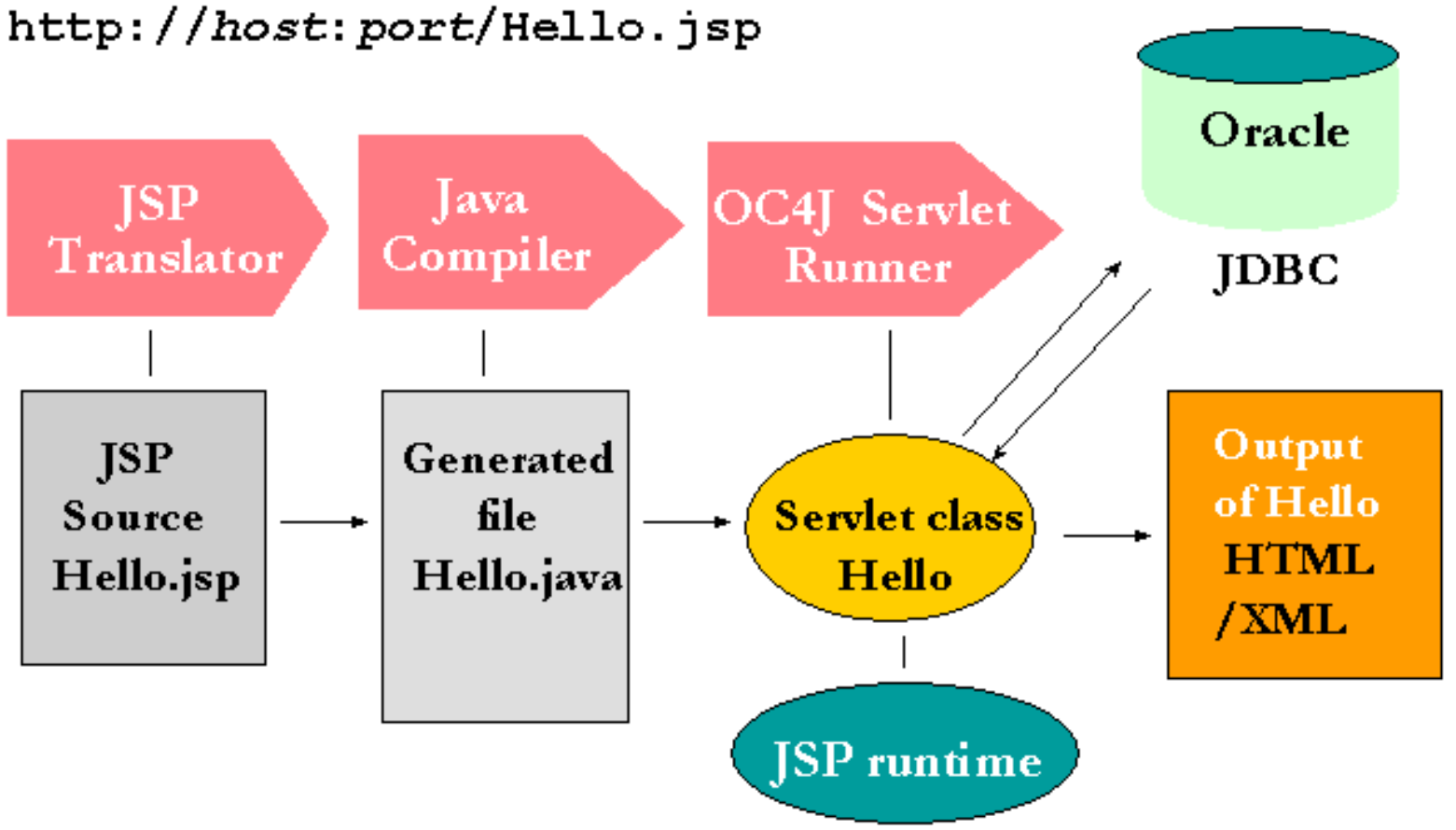
Carregando uma página JSP

- Na requisição, uma página JSP passa pelas seguintes etapas:
  - O *JSP engine* converte a página JSP em um servlet (arquivo fonte da classe servlet Java)
  - Em seguida, a classe servlet gerada é automaticamente compilada
  - O *servlet container* inicializa e executa o servlet
  - O servlet então gera o conteúdo HTML da resposta
  - O servlet container envia a resposta para o cliente

- No Tomcat, os arquivos fonte e .class do servlet gerado podem ser encontrados em:
  - diretorio_tomcat\work\Catalina\localhost\nome_aplicacao\org\apache\jsp

Carregando uma página JSP

How is a JSP Served ?



Tags JSP

➢No JSP há 5 tipos de tags que podem ser inseridas em uma página HTML para gerar conteúdo dinâmico, cada uma delas com um formato de delimitador distinto:

- **Declaração:** permite a declaração de variáveis e métodos Sintaxe: <%! *declaração* %>
- **Expressão:** apresenta algum resultado. A expressão é avaliada e o resultado é inserido no documento HTML no lugar da tag JSP. Sintaxe: <%= *expressão* %>
- ***Scriptlet*:** permite incluir blocos de código Java (múltiplas linhas, ao contrário de expressão, que pode conter uma única expressão Java. Sintaxe: <% *código Java* %>
- **Diretiva:** permite incluir informações sobre a página (há 3 tipos de diretivas (page, include, taglib). Sintaxe: <%@ *diretiva* atributo=valor %>
- **Ação:** permite realizar uma ação como, acessar e modificar JavaBeans, redirecionar a requisição para outra página, etc.
- Sintaxe: <jsp:acao atributo=valor />

Declaração

➢**Declaração:** permite a declaração de variáveis e métodos Java

➢Sintaxe: `<%! declaração %>`

➢Exemplo:

➢`<%! String nome = “Joao”; %>`

➢Declarando um método:

```
➢<%! int soma(int a, int b) {
        return (a + b); }
➢%>
```

Expressão

- Uma expressão JSP contém código Java que resulta em um valor que pode ser escrito na página HTML
- Sintaxe: `<%= valor %>`
- O valor pode ser qualquer coisa que possa ser convertido em uma String Java (inclusive valores primitivos, objetos, chamadas de métodos que retornem algum valor)
- No caso de objetos, somente aqueles que tem uma implementação útil do método *toString()* podem oferecer de fato informações úteis (a versão do método padrão *toString()* da classe Object mostra somente o nome da classe e o hash code)
- No exemplo, o objeto Date possui o método *toString()* para mostrar a data e a hora de sua criação:

```
<%= new java.util.Date() %>   Obs.: sem ponto-e-vírgula!
```

- Isto é convertido em: (código na classe servlet)

```
out.println(new java.util.Date());
```

Diretivas

- Diretivas permitem incluir instruções especiais que podem ser passadas para o compilador JSP no momento da conversão da página
- Existem três tipos de diretivas:
  - page: informações sobre a página (atributos como *import, language*, *contentType, errorPage,* etc.)
  - include: inclusão de arquivos
  - taglib: declara uma biblioteca de tags a ser utilizada
- Para importar pacotes ou classes Java é utilizada a diretiva *page*
- Sintaxe:

```
<%@ page import="nome_pacote ou nome_classe" %>
```

- Exemplo: importando um pacote:

```
<%@ page import="java.util.*" %>
```

Obs.: sem ponto-e-vírgula!

- Exemplo: importando uma classe:

```
<%@ page import="java.util.Date" %>
```

  - Dessa forma, é preciso apenas utilizar o nome da classe:
  - `<%= new Date() %>`

Scriptlet

- *Scriptlet* permite a inclusão de várias linhas (bloco) de código Java em uma página JSP
- Sintaxe: `<% código Java %>`
- Exemplo: usando o comando if-else:

```
<%
    int a = 10;
    if(a >= 5) {
      out.println("é maior ou igual a 5");
  Obs.: ponto-e-vírgula no final!
     }
    else {
      out.println("é menor que 5");
    }
%>
```

Objetos implícitos

- Alguns objetos são criados pelo JSP engine na criação do servlet e podem ser acessados nas páginas JSP:
  - *request:* objeto *HttpServletRequest* para acessar as informações da requisição HTTP
  - *response:* objeto *HttpServletResponse* para acessar as informações da resposta HTTP
  - *out:* objeto *JSPWriter* para escrever um conteúdo no response
  - *session:* objeto *HttpSession* que representa a sessão de um usuário
- Exemplo: um campo de texto (o valor é enviado na requisição)

```
<input type="text" name="nome" value="" size="25" />
```

- Pode-se obter o parâmetro da requisição utilizando:

```
<%
   String nomeUsuario = request.getParameter("nome");
%>
```

Exemplo JSP

➢Exemplo: enviando parâmetros da requisição **(index.jsp)**

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1">
<title>Parâmetro da requisição</title>
</head>
<body>
    <h2>Pagina de Teste JSP</h2>
    <form action="resposta.jsp" method="get">
    Digite o seu nome: <br />
    <input type="text" name="nome" value="" size="25" />
    <input type="submit" value="Submeter" />
    </form>
</body>
</html>
```

Exemplo JSP

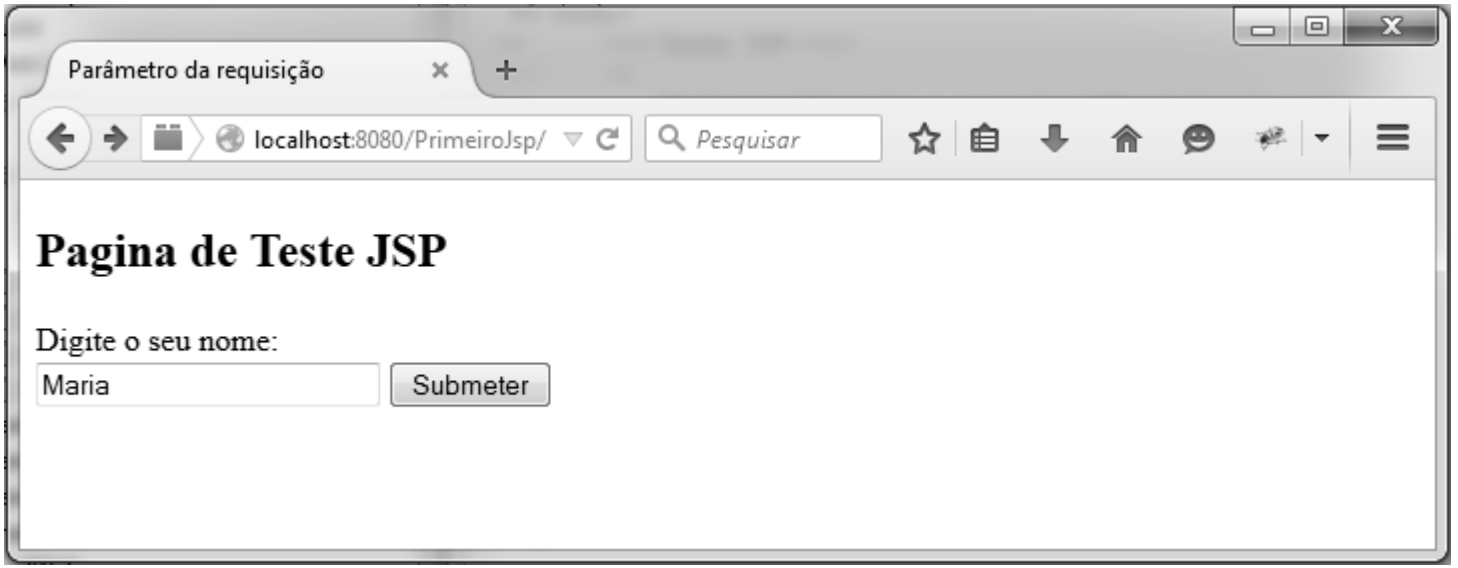
➢Exemplo: enviando parâmetros da requisição **(resposta.jsp)**

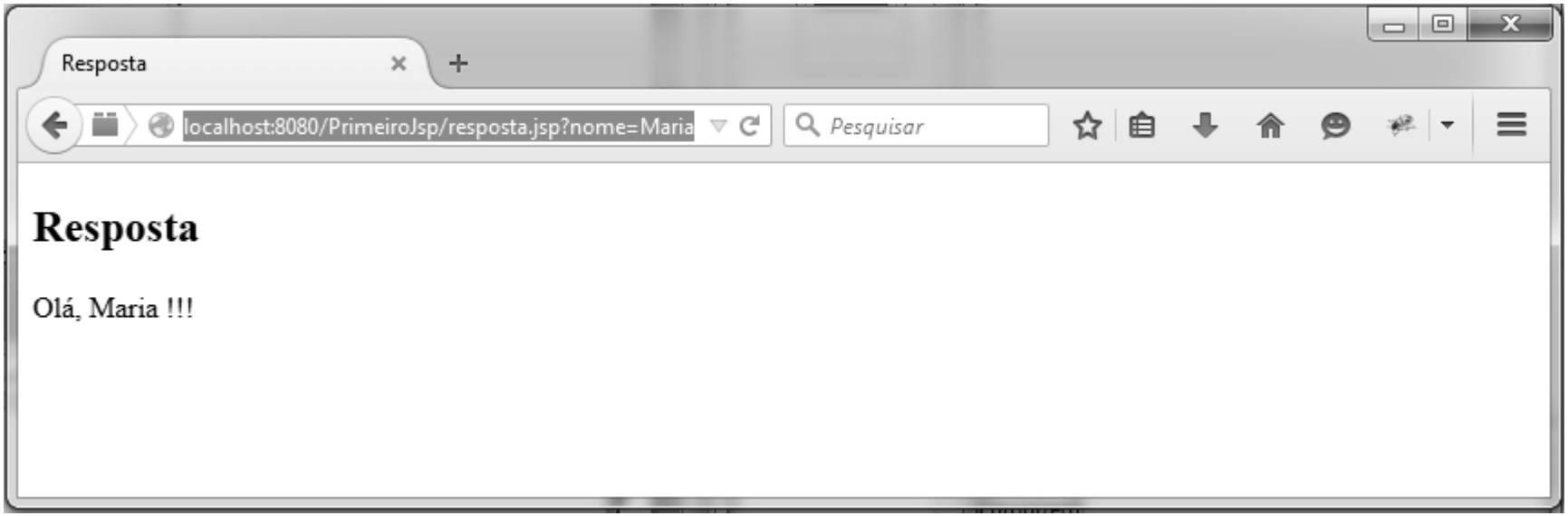
```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1">
<title>Resposta</title>
</head>
<body>
    <h2>Resposta</h2>
    <%
      String nomeUsuario = request.getParameter("nome");
    %>

    Olá, <%= nomeUsuario %> !!!
</body>
</html>
```

Objeto implícito
HttpServletRequest

Exemplo JSP

- Exemplo: enviando parâmetros da requisição





Ações

- As ações basicamente, possuem duas funções importantes:
  - Permitem a transferência de controle entre páginas (redirecionar uma requisição para outra página)
  - Permitem a interação das páginas JSP com JavaBeans (acessar, modificar JavaBeans no servidor)

- Para redirecionar uma requisição para uma outra página:
  <jsp:forward page="algumaURL" />
  - Exemplo:

```
<%
   int numero = request.getParameter("valor");
   if(numero > 0) { %>
     <jsp:forward page="calcula.jsp" />                    <%
  }
   else { %>
     <jsp:forward page="erro.jsp" />
<%   }
%>
```

Ações

- É possível também incorporar um outro arquivo na página:
  <jsp:include page=”algumaURL” />

- Exemplo:

```
<body>
  <jsp:include page="cabecalho.jsp"/>
  <form action="principal.jsp" method="post">
    Nome do aluno:
    <input type="text" name="nomeAluno" /><br><br>
    <input type="submit" value="Ok" />
  </form>
  <jsp:include page="rodape.jsp"/>
</body>
```

Referências

- D. Fields, Mark Kolb. Desenvolvendo na Web com Java Server Pages. Ed. Ciência Moderna.
- Bryan Basham, Kathy Sierra, Bert Bates. Servlets e JSP. Alta Books Editora, 2010.
- Tutoriais do Java EE (site da Oracle):
- http://www.oracle.com/technetwork/java/javaee/documentation/tutorials-137605.html

MC0037 – Programação para Web

Aula 5: JSTL: JSP Standard Tag Library

Aula de hoje: Roteiro

- JSTL – JSP Standard Tag Library

JSTL

- JSTL: JSP Standard Tag Library
- É uma **biblioteca de tags**: oferece um conjunto padrão de tags que pode ser utilizado em qualquer servidor de aplicação Java
- Reduz a necessidade de misturar código Java (*scriptlets*) com tags

- Há 4 categorias de bibliotecas:
  - Core
  - Database access
  - XML processing
  - Internationalization and formatting

- Juntamente com a JSTL, há a EL (Expression Language), que começou como parte da JSTL mas agora tornou-se um componente padrão do JSP

Expression Language (EL)

➢A EL é uma linguagem que foi criada com o intuito de facilitar o gerenciamento dos dados da aplicação web

➢Os desenvolvedores web (front end) não precisam conhecer a linguagem de implementação da aplicação

➢Oferece uma maneira de acessar objetos Java (JavaBeans) e seus atributos sem utilizar a linguagem Java

➢As expressões EL devem estar entre chaves e iniciar com o caracter $:

➢**`${expressão}`**

➢A expressão deve ter algum resultado, pode ser o valor de uma variável ou o resultado de uma expressão aritmética ou relacional (lógica)

Parâmetros da requisição na EL

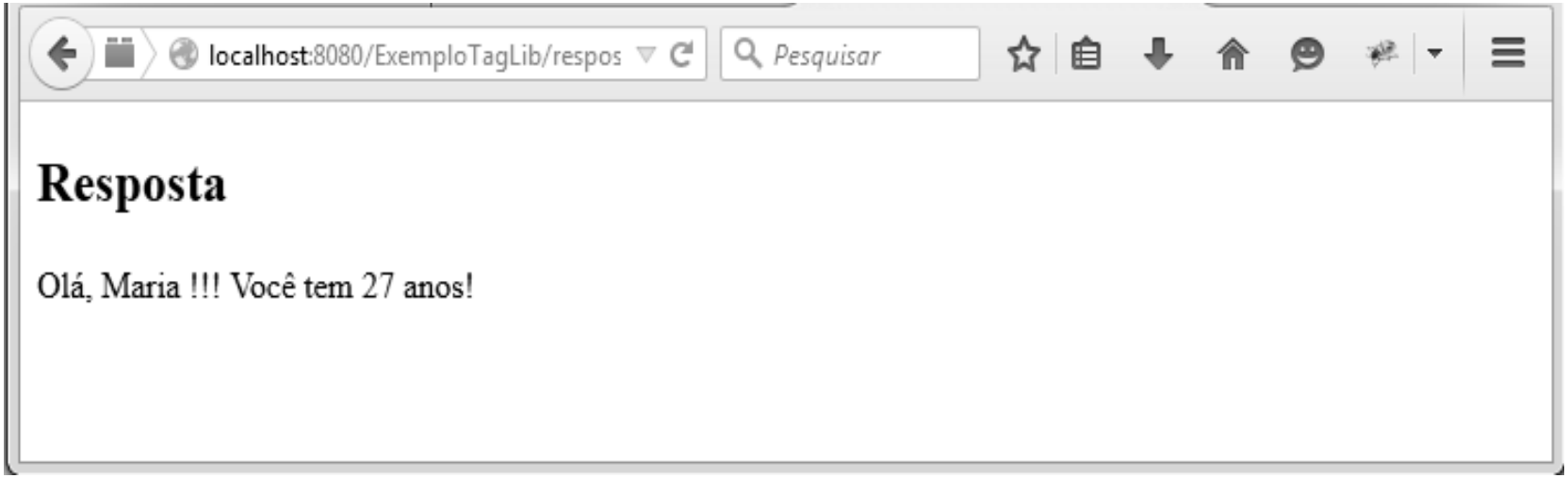
```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
	<h2>Pagina de Teste TagLib</h2>
	<form action="/salvar" method="post">
		Digite o seu nome: <br>
		<input type="text" name="nome" value="" size="25"/>
		<br>
		Digite o ano de nascimento: <br>
		<input type="text" name="ano" value="" size="25" />
		<br>
		<input type="submit" value="Submeter" />
	</form>
</body>
</html>
```

Parâmetros da requisição na EL

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"
<title>Resposta</title>
</head>
<body>
        <h2>Resposta</h2>

        <c:set var="nomeUsuario" value="${param.nome}" />
        <c:set var="anoNasc" value="${param.ano}" />
        <c:set var="anoAtual" value="2015" />
        <c:set var="idade" value="${anoAtual - anoNasc}" />

        Olá, <c:out value="${nomeUsuario}" /> !!!
        Você tem ${idade} anos!
</body>
</html>
```

Objeto implícito da EL

Parâmetros da requisição na EL



Alguns operadores EL

Operadores Aritméticos:

Adição	+
Subtração	-
Multiplicação	*
Divisão	/ ou div
Módulo	% ou mod

Operadores lógicos:

AND	&& ou **and**
OR	|| ou **or**
NOT	! ou **not**

Operadores relacionais:

Igual	== ou **eq**
Diferente	**!=** ou **ne**
Menor	< ou **lt**
Maior	> ou **gt**
Menor ou igual	<= ou **le**
Maior ou igual	>= ou **ge**

Algumas palavras reservadas:

- true, false, empty

Utilizando JSTL

- Para utilizar as tags JSTL, é preciso incluir as bibliotecas do JSTL no gradle.
  - compile('org.springframework.boot:spring-boot-starter-web')

- Também é preciso incluir a seguinte diretiva para que o JSP possa referenciar as bibliotecas:

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

- Neste exemplo, a URI corresponde à biblioteca core e o prefixo “c” é uma abreviação para referenciar as tags dessa biblioteca.

Exemplos de tags

➢Tag out: para apresentar um resultado:

```
<c:out value="valor" />
```

Exemplos:

```
<c:out value="Primeira mensagem!" />
<c:out value="${aluno.nome}" />
```

➢Tag if: testa uma condição (usado quando não há else):

```
➢<c:if test="condicao">
        // faz alguma coisa
    </c:if>
```

Exemplo:

```
<c:if test="${!empty aluno.nome}">
    Nome: ${aluno.nome}
</c:if>
```

Exemplos de tags

```
<label>Situação:

<c:if test="${cc.ativa == true}">
        <input type="radio" name="ativa" value="true" checked
        /> Ativa <br />
        <input type="radio" name="ativa" value="false" /> Inativa
        <br />
</c:if>

<c:if test="${cc.ativa != true}">
         <input type="radio" name="ativa" value="true" /> Ativa
         <br />
         <input type="radio" name="ativa" value="false"
         checked/> Inativa <br />

</c:if>
```

Exemplos de tags

➢Tag choose: utilizado quando há várias condições (é equivalente à estrutura if...else do Java, mas também é semelhante ao switch-case):

```
<c:choose>
   <c:when test="condicao1">
      // acao 1
   </c:when>
   <c:when test="condicao2">
      // acao 2
   </c:when>
   <c:otherwise>
      // outra acao
   </c:otherwise>
</c:choose>
```

Exemplos de tags

- Tag for-each: utilizado para iterações:

```
<c:forEach var="nomeVariavel" items="listaItens">
   // ...
</c:forEach>
```

- Exemplo:

```
<c:forEach var="aluno" items="${dao.lista}" >
   Nome: ${aluno.nome}<br>
   Email: ${aluno.email}<br>
   Endereço: ${aluno.endereco}<br>
</c:forEach>
```

Exemplos de tags

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Listar Contas Correntes</title>
</head>
<body>

<h3>Contas Correntes</h3>
<br>
<table border="1">

<thead>
<tr>
<th><b>Número</b></th>
<th><b>Agencia</b></th>
<th><b>Descricao</b></th>
<!-- <th><b>Transactions</b></th> -->
</tr>
</thead>
<tbody>
<c:forEach items="${ccs}" var="cc">
<tr>
<td><c:out value="${cc.numero}"></c:out></td>
<td><c:out value="${cc.agencia}"></c:out></td>
<td><c:out value="${cc.descricao}"></c:out></td>

<td><a href="/edit/${cc.id}">
<button type="submit" class="btn btn-primary">Editar CC</button>
</a></td>
<%--
<td><a href="/index/${lou.id}/delete">
<button type="submit" class="btn btn-primary">Delete
User</button>
</a></td> --%>
</tr>

</c:forEach>
</tbody>
</table>

<a href="/new">Nova CC</a>

</body>
</html>
```

Exemplos de tags

- Tag set: declarar variáveis:

```
<c:set var="nomeVariavel" value="valor" />
```

- Exemplos:

```
<c:set var="nomeUsuario" value="Joao" />
<c:set var="anoNasc" value="1980" />
<c:set var="idade" value="${param.idade}" />
```

Taglibs Form do Spring

➢Configuração

- `<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>`
- Deve ter a dependência: spring-webmvc.jar
  - compile('org.springframework.boot:spring-boot-starter-web')
- https://docs.spring.io/spring/docs/3.2.x/spring-framework-reference/html/view.html

Taglibs Form do Spring

- Tags: Form
  - `<form:form>`
- Tags: Input
  - `<form:input type=“date” path="firstName" />`
- Tags: Password
  - `<form:password path=“password" />`
- Tags: Checkbox
  - `<form:checkbox path=“situacao" />`

Taglibs Form do Spring

➢Tags: Checkbox

- `<form:checkbox path=“situacao" items=“${situacoes}”/>`

➢Tags: Radio

- `<form:radiobutton path="sex" value="M"/>`
- `<form:radiobuttons items="${jobItem}" path="job" />`

➢Tags: Select

- `<form:select path="country" items="${countryItems}" />`

Taglibs Form do Spring

```
<form:form action="/save" modelAttribute="cc">
<label>Número</label><form:input path="numero"/> <br/>
<label>Agência</label><form:input path="agencia"/> <br/>
<label>Descrição</label><form:input path="descricao"/> <br/>
<label>Situação</label><form:radiobutton path="ativa"
value="true"/> Ativa
<form:radiobutton path="ativa" value="false"/> Inativa <br/>
<label>Variação</label><form:input path="variacao"/> <br/>
<form:hidden path="id"/> <br/>
<input type="submit" name="action" value="Salvar conta" />
</form:form>
```

Referências

- D. Fields, Mark Kolb. Desenvolvendo na Web com Java Server Pages. Ed. Ciência Moderna.
- Bryan Basham, Kathy Sierra, Bert Bates. Servlets e JSP. Alta Books Editora, 2010.
- Tutoriais do Java EE (site da Oracle):
  - http://www.oracle.com/technetwork/java/javaee/documentation/tutorials-137605.html
  - http://docs.oracle.com/javaee/6/tutorial/doc/gjddd.html
- JSP Standard Tag Library:
  - https://jstl.java.net/