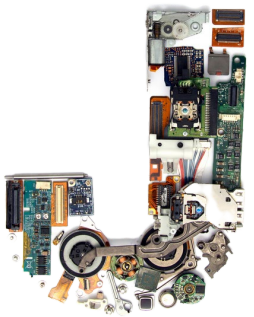
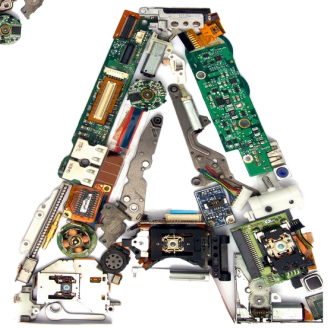
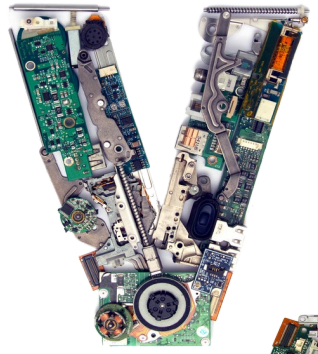
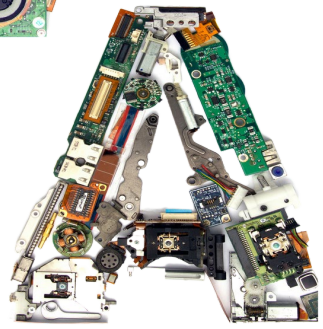




Programação Orientada a Objetos com Java e WEB



JSTL – JavaServer Pages Standard Tag Library



O que é JSTL?

JSTL é o acrônimo de **JavaServer Pages Standard Tag Library.**

São usados para recuperar dados de forma transparente usando como componente básico da JEE o qual é muito usado na programação pura (quando programamos diretamente e tão somente no JSP **(Java Server Pages))**.

Ao usar JSTL podemos usar de maneira **embutida**, o código de lógica Java, sem necessariamente usar uma classe Java. JSTL permite que os programadores JSP usem tags em vez de código Java.

O uso do JSTL é bastante aceito e sugerido quando necessitamos de algo mais rápido, uma simples amostragem, como exemplos rápidos, consultas pré-ordenadas.

Exemplo do uso de scriptlet

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Uso de Scriptlets</title>
</head>
<body>
    <% for(int j = 1; j <= 10; j++) { %>
        <%= j %> <br/>
    <% } %>
</body>
</html>
```

uso abusivo de scriptlet deixa o código confuso e difícil de manutenção

O que é JSTL?

Em resumo:

Uma Taglib nada mais é que uma biblioteca de tags customizadas que são utilizadas na composição de páginas JSP. Em um passo adiante, pode-se dizer que uma Taglib é uma biblioteca de “classes Java” que são utilizadas “na forma de tags” para auxiliar na geração de conteúdo dinâmico em uma página JSP.

Outro recurso muito importante, introduzido pela **JSTL**, é a **EL**, ou seja, **Expression Language**(*Linguagem de Expressão*), que é uma forma mais simples de acessar e manipular objetos, sua sintaxe, é composta por um “cifrão”, seguido por um par de chaves, contendo o nome do objeto, por exemplo:

${objeto.nomeAtributo}

expression language (EL)

Biblioteca JSTL

A biblioteca do JSTL está distribuída em **5**(cinco) pacotes, agrupados por funcionalidade:

Pacote	Sugestão de prefixo	Descrição
JSTL core	c	Tags relacionadas à lógica e controle como (**if**, **forEach**, **url**, **set**, **import**, etc.).
JSTL fmt	fmt	Tags para formatação e internacionalização de dados
JSTL sql	sql	Tags para acessar, inserir, alterar e deletar dados em um servidor de banco de dados.
JSTL xml	xml	Tags para seleção, parser e transformação de modelos XML.
JSTL functions	fn	Conjunto de funções para o processamento de objetos Strings e coleções.

Configuração da biblioteca JSTL

Sempre que adicionamos uma taglib devemos ter os **jars** equivalentes em seu diretório **WEB-INF/lib**.

Arquivo **jar** com a implementação core da taglib → **jstl-1.2.jar**.

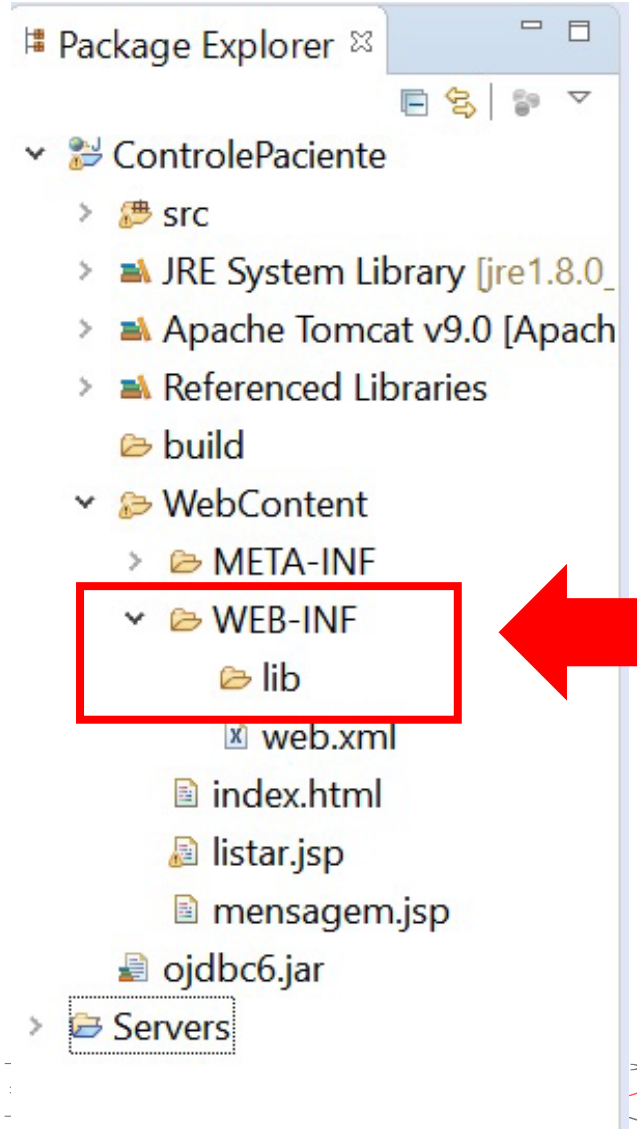
É importante observar que cada pacote, possui seu próprio endereço de **URI** (*Uniforme Resource Identifier*), que deverá ser inserido como atributo da tag jsp **<%@ taglib />** seguido pela sugestão de prefixo. Por exemplo:

<%@ taglib uri="uri_da_taglib**" prefix="**prefixo**" %>**

Para a biblioteca core:

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

Configuração da biblioteca JSTL



Copiar o driver jstl-1-2.jar para a pasta lib dentro de WEB-INF

jstl-1.2.jar

Configuração da biblioteca JSTL

É importante, observar, que uma vez declarado o pacote específico, você irá invocar as **tags** usando o prefixo definido no atributo **prefix**, conforme o exemplo abaixo:

<prefixo:nomeTag atributo="...">

corpo da tag ...

</prefixo:nomeTag>

Principais Tags

Tag **<c:out>** é utilizada para exibir o resultado de uma expressão. É bastante similar ao uso do scriptlet **<%= ...>**, mas a tag **<c:out>** permite o uso do operador ponto para acessar as propriedades dos objetos. Para imprimir um valor: **<c:out value = "cliente.rua"/>**. Exemplo:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <c:out value = "${'<tag> , &'}"/>
</body>
</html>
```

import da taglib

Principais Tags

Tag **<c:set>** configura o valor de uma variável. Exemplo:

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

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <c:set var = "salary" scope = "session" value = "${2000*2}"/>
    <c:out value = "${salary}"/>
</body>
</html>
```

Principais Tags

Tag **<c:forEach>** é utilizada para executar uma estrutura de repetição do tipo **for**, **while** ou **do-while.** A tag é bastante utilizada para iterar sobre coleções de objetos.

Atributos da tag **<c:forEach>:**

Atributo	Descrição	Requerido	Padrão
items	Informação para iterar	Não	Nenhum
begin	Define valor inicial (0 = primeiro, 1 = segundo, ...)	Não	0
end	Define valor final (0 = primeiro, 1 = segundo, ...)	Não	Último elemento
step	Incremento ou decremento a cada iteração	Não	1
var	Nome da variável que será utilizada na tag	Não	Nenhum
varStatus	Nome da variável para mostrar informações da iteração	Não	Nenhum

Tag <c:forEach> - exemplo

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<c:forEach var="i" begin="1" end="5">
    Valor <c:out value = "${i}"/><p>
</c:forEach>
</body>
</html>
```

import da taglib

Principais Tags

Tag **<c:if>** é utilizada para avaliar o seu conteúdo e, caso seja verdadeiro, exibir o conteúdo do corpo da tag. Exemplo:

```
<c:set var = "salary" scope = "session" value = "${2000*2}"/>
    <c:if test = "${salary > 2000}">
         <p>Meu salário é:  <c:out value = "${salary}"/><p>
    </c:if>


<c:if test="${empty lista}">
        <h1>Nenhuma Empresa Cadastrada</h1>
</c:if>
```

Formatação de datas

A JSTL possui um outro pacote, focado nesse tipo de formatação. Esse é o pacote "**fmt**". Ele contém uma **taglib** chamada "**formatDate**", que faz isso para nós. Ela recebe o padrão que deve formatar a data (por exemplo, **dd/MM/yyyy** que representa dia/mês/ano) e a data a ser exibida.

Veja o código abaixo:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<td>
        <fmt:formatDate pattern="dd/MM/yyyy" value="${p.data}" />
</td>
```

Formatação de datas

Além das datas, podemos formatar também números. Podemos usar a **taglib** "**formatNumber**" para formatar o número:

```
<fmt:formatNumber value="${p.preco}" type="currency"/>
```

Referências

- DEITEL, H. M., DEITEL, P. J. **JAVA como programar**. 10ª edição. São Paulo: Prentice-Hall, 2010.