

# JSP Standard Tag Library (JSTL) e Expression Language (EL)

Laboratório de Programação (5COP011)  
Prof. Bruno Bogaz Zarpelão

Departamento de Computação - 2016



# Objetivo

- Discutir formas de não utilizar *scriptlets* em páginas JSP combinadas com Servlets.
- Tópicos:
  - JSP Standard Tag Library (JSTL).
  - Expression Language (EL).

# JSTL

- A JSTL é uma biblioteca que traz vários *action elements* (também chamados de *tags*) úteis para o dia a dia de um programador web;
- Temos, por exemplo, tags para processamento de XML, acesso a base de dados e tarefas básicas como desvios condicionais e loops;
- Com estas tags, conseguimos utilizar código *HTML-like* para invocar bibliotecas escritas em Java;

# JSP Action Element

- *JSP action elements* representam ações dinâmicas que são processadas no momento de execução da página;
- As diretivas (“<@ include...”, “<@ page...”), por exemplo, não são processadas no momento de execução. Elas são processadas na fase de tradução do JSP para Servlet;


# JSP Action Element

```
<%@ page contentType="text/html" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core" %>
<html>
  <head>
    <title>JSP is Easy</title>
  </head>
  <body bgcolor="white">

    <h1>JSP is as easy as ...</h1>

    <!-- Calculate the sum of 1 + 2 + 3 dynamically -->
    1 + 2 + 3 = <c:out value="${1 + 2 + 3}" />

  </body>
</html>
```



# JSTL

- As tags JSTL são organizadas nas seguintes áreas:

Área	Prefixo	Descrição	URI
Core	"c"	Lógica e controle (if, forEach)	<a href="http://java.sun.com/jsp/jstl/core">http://java.sun.com/jsp/jstl/core</a>
XML	"x"	Manipulação de XML	<a href="http://java.sun.com/jsp/jstl/core/xml">http://java.sun.com/jsp/jstl/core/xml</a>
I18N	"fmt"	Formatação e internacionalização	<a href="http://java.sun.com/jsp/jstl/fmt">http://java.sun.com/jsp/jstl/fmt</a>
Database	"sql"	Acesso a banco de dados	<a href="http://java.sun.com/jsp/jstl/sql">http://java.sun.com/jsp/jstl/sql</a>
Functions	"fn"	Processamento de Strings e Collections	<a href="http://java.sun.com/jsp/jstl/functions">http://java.sun.com/jsp/jstl/functions</a>

# JSTL

- Para usar a JSTL:
  1. Adicionar as bibliotecas necessárias em WEB-INF/lib;
  2. No cabeçalho da página JSP, declarar a utilização da tag. Exemplo: `<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>`

# JSTL Expression Language

- A EL (Expression Language) é inspirada no JavaScript;
- Com a EL, é possível atribuir valores para atributos de action elements a partir de diferentes fontes de dados;
- Duas de suas vantagens: lida bem com atributos cujo valor é *null* e realiza uma série de conversões automáticas de tipos de dados;



# Expression Language

- Uma expressão EL começa com “\${” ou “#{” e termina com “}”;
- Dentro da expressão você pode usar números, variáveis implícitas que deem acesso a dados da requisição HTTP, dados da aplicação, etc...
- Objetivo é evitar ao máximo o uso de *scriptlets*;
- JSTL e EL são HTML-like. Logo, são mais adequadas para construir páginas web do que os *scriptlets*;

# Vamos analisar uma coisa...

- Percebam como temos a evolução do desenvolvimento Java com WEB:
  - Servlets: usavam só código Java, inclusive para gerar respostas em HTML;
  - JSP: a grosso modo, misturavam HTML com Java;
  - JSP com JSTL e EL: permitem aproveitar o potencial do Java usando tags e comandos que não são tão diferentes de HTML;
- O intuito é não misturar Java com HTML...vamos evitar o uso de scriptlets!

# Expression Language

- Podemos acessar propriedades de objetos Java com EL (a classe deve seguir o formato JavaBean):
  - `<%jsp:useBean id="contato" class="pacote.Contato" %>`
  - `${contato.nome}` ou `${contato["nome"]}`;
- Podemos acessar um vetor: “indice” tem que ser do tipo *int*
  - `${contato.listaTelefones[1]}` ou `${contato.listaTelefones.indice}`

# JavaBeans

- A especificação JavaBeans define uma série de regras de programação para classes Java de forma a torná-las mais fáceis de serem usadas;
- Os JavaBeans podem ser construídos e acessados por servlets, que depois se encarregam de chamar os JSPs;
- Entretanto, houve uma preocupação em desenvolver um suporte direto a JavaBeans no JSP, para facilitar a vida de designers web;

# Convenções de um *bean*

- Sempre deve estar associado a um pacote;
- Sempre deve ter um construtor sem parâmetros. Desta forma, é possível criar um objeto para este bean conhecendo apenas o nome da classe;
- Os atributos dos JavaBeans devem ser acessados por meio de *getters* e *setters*;
- O *bean* deve implementar a interface `java.io.Serializable` ou a interface `java.io.Externalizable`;

# Convenções de um *bean*

- A interface `Serializable` vai permitir que este objeto seja transformado em um *stream* de bytes, possibilitando:
  - A transmissão dele pela rede;
  - A persistência dele em um arquivo;
- Os dados carregados em um *beans* são nomeados propriedades (*properties*);
- No código JSP, os *beans* são criados por meio de *action elements*;



Ex.:  
Cookie!!!!

# Expression Language

- Também podemos trabalhar com valores literais:
  - $\${\text{contato.idade} + 10}$
  - $\${45}$
  - $\${\text{"UEL"}}$

# Expression Language

- Operadores:
  - Aritméticos: +, - (binário), \*, / e div, % e mod, - (unário);
  - Lógico: and, &&, or, ||, not, !;
  - Relacional: ==, eq, !=, ne, <, lt, >, gt, <=, ge, >=, le;
  - Operador empty para testar se o valor é null ou está vazio;



# Expression Language: variáveis implícitas

Nome da variável	Descrição
pageScope	Permite acesso às variáveis no escopo da página atual
requestScope	Permite acesso às variáveis no escopo do request
sessionScope	Permite acesso às variáveis no escopo da sessão
applicationScope	Permite acesso às variáveis no escopo da aplicação
param	Acesso ao valor do parâmetro do request

Exemplo: `${param["nome"]}` -> acessa o valor do parâmetro nome presente no request

# Expression Language

- Vimos várias expressões que começam com “\${”;
- Uma expressão também pode ser escrita iniciando por “#{”. Neste caso, podemos passar valores para a expressão:
  - Método em classe Java “public List obterLista(int i)”;
  - Podemos acessar em EL: #{objeto.obterLista(10)};

# JSTL

- Vamos voltar ao JSTL!
- JSTL core, if.
- JSTL core, forEach
- JSTL core, choose.
- JSTL fmt, formatDate.

# c:if

```
<c:if test="\${salary > 2000}">  
  <p>My salary is: <c:out value="\${salary}"/><p>  
</c:if>
```

# c:forEach

```
<c:forEach var="i" begin="1" end="5">  
  Item <c:out value="${i}"/><p>  
</c:forEach>
```

OU

```
<c:forEach var="contato" items="${dao.lista}">  
  <tr>  
    <td>${contato.nome}</td>  
    <td>${contato.endereco}</td>  
  </tr>  
</c:forEach>
```

# c:choose

```
<c:choose>  
  <c:when test="${not empty contato.email}">  
    <a href="mailto:${contato.email}">${contato.email}</a>  
  </c:when>  
  <c:otherwise> E-mail não informado </c:otherwise>  
</c:choose>
```

# fmt:formatDate

```
<fmt:formatDate  
  value="${contato.dataNascimento.time}"  
  pattern="dd/MM/yyyy"  
>
```

# JSTL

- JSTL core, set:
  - O set é utilizado para criar e atribuir valor a uma variável;
  - Suponhamos que queremos mostrar um contador com o número de acessos do usuário à página;
  - Podemos utilizar o set juntamente com o escopo de sessão;

```
<c:set var="salary" scope="session" value="${2000*2}"/>
```



# Exercício

- Fazer aplicação para cadastro de produtos (com carrinho de compras) utilizando JSP, JSTL e Servlets. Não deve-se utilizar scriptlets.

# Bibliografia

- H. Bergsten. “Java Server Pages”, O’Reilly Media, 2. ed., 2002;
- Java EE 6 Tutorial – Expression Language. Disponível em:  
<http://docs.oracle.com/javaee/6/tutorial/doc/gjddd.html>