

A Plataforma Java

Programação Web com Java



Uso de JSTL

- Taglibs Simples -



Prof. Giuliano Prado de Moraes Giglio, M.Sc.





O JSP sem padrão

- É muito comum!
- Scriptlets espalhados em várias páginas
- Código confuso = manutenção +trabalhosa
- Regras de negócio não deveriam ser tratadas dentro do JSP (foge ao padrão MVC!)
- JSP deve ser usado somente para a camada de visualização (viewer), somente exibindo o conteúdo dos objetos e nada de scriptlets



JSTL: o JSP com padrão

■ HISTÓRICO

- **Julho/2001** - Criado o projeto (JSR-052)
- **Junho/2002** – lançado o JSTL 1.0 baseado no JSP 1.2 (Tomcat4 e maioria dos servidores corporativos)
- **Janeiro/2004** - lançado o JSTL 1.1 baseado no JSP 2.0 (Tomcat 5)
- **Julho/2004** - lançado o JSTL 1.1.1



JSTL: o JSP com padrão

- Padronizar as aplicações JSP !
- Dar soluções fáceis de usar para tarefas mais comuns

JSTL: o JSP com padrão

Tipos de soluções JSTL 1.0

Tipo	URI	Prefixo	Exemplo
Core	http://java.sun.com/jstl/core	c	<c:nomeDaTag...>
XML	http://java.sun.com/jstl/xml	x	<x:nomeDaTag...>
Internationalization	http://java.sun.com/jstl/fmt	fmt	<fmt:nomeDaTag...>
Database	http://java.sun.com/jstl/sql	sql	<sql:nomeDaTag...>

JSTL: o JSP com padrão

Tipos de soluções JSTL 1.1

Tipo	URI	Prefixo	Exemplo
Core	http://java.sun.com/jsp/jstl/core	c	<c:nomeDaTag...>
XML	http://java.sun.com/jsp/jstl/xml	x	<x:nomeDaTag...>
l18n	http://java.sun.com/jsp/jstl/fmt	fmt	<fmt:nomeDaTag...>
Database	http://java.sun.com/jsp/jstl/sql	sql	<sql:nomeDaTag...>
Funções	http://java.sun.com/jsp/jstl/functions	fn	<fn:nomeDaTag...>



Como instalar o JSTL?

■ 1. Faça o download:

<http://www.apache.org/dist/jakarta/taglibs/standard/>

■ 2. Descompacte o arquivo e copie:

a) `/jakarta-taglibs-standard-1.*/tld/*` para **WEB-INF**

b) `/jakarta-taglibs-standard-1.*/lib/*` para **WEB-INF/lib**

Como instalar o JSTL?

3. Adicione essas informações no **web.xml**:

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/fmt</taglib-uri>  
  <taglib-location>/WEB-INF/fmt.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/fmt-rt</taglib-uri>  
  <taglib-location>/WEB-INF/fmt-rt.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/core</taglib-uri>  
  <taglib-location>/WEB-INF/c.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/core-rt</taglib-uri>  
  <taglib-location>/WEB-INF/c-rt.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/sql</taglib-uri>  
  <taglib-location>/WEB-INF/sql.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/sql-rt</taglib-uri>  
  <taglib-location>/WEB-INF/sql-rt.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/x</taglib-uri>  
  <taglib-location>/WEB-INF/x.tld</taglib-location>  
</taglib>
```

```
<taglib>  
  <taglib-uri>http://java.sun.com/jstl/x-rt</taglib-uri>  
  <taglib-location>/WEB-INF/x-rt.tld</taglib-location>  
</taglib>
```


Como instalar o JSTL?

- 4. Na sua página JSP declare os tipos que for utilizar:

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

- 5. Adicione a biblioteca JSTL em seu projeto

- 6. Depois é só sair usando!

```
<H1><c:out value="Conexão Java 2004" /></H1>
```

Sintaxe

- Não importa onde a Expressão Lógica é usada, ela sempre é invocada de uma maneira consistente:

- `${exp}` ou `#{exp}`

Obs.: Na plataforma J2EE não podem ser usadas em conjunto.

- Exemplos:

- `${true}`
- `${ "aqui jaz um texto" }`
- `${5*4} <!-- resulta no numero 20--%>`



Entendendo o JSTL

- O JSTL é uma coleção de quatro bibliotecas tags.
- Cada biblioteca de tags fornece ações úteis (ou tags) baseados nas seguintes áreas funcionais:
 - Core
 - Internacionalização (I18n) e formatação
 - Acesso a banco de dados relacional (tags SQL)
 - Processamento de XML (tags XML)



A Core Tag Library

- Contém um centro de ações de propósito geral, que fornecem soluções simples, mas efetivas, a problemas comuns que os desenvolvedores experimentam em quase toda aplicação JSP.
- Grupo de tags mais usadas freqüentemente:
 - **<c:if />** para condições
 - **<c:forEach />** para interação
 - **<c:choose /> ... <c:when />... <c:otherwise />** para um fluxo seletivo
 - **<c:set />** e **<c:remove />** para trabalhar com escopo de variáveis
 - **<c:out />** para fazer a saída de valores de variáveis e expressões
 - **<c:catch />** para trabalhar com exceções Java
 - **<c:url />** para criar e trabalhar com URL's

Exibindo Objetos

- O formato padrão é:
`<c:out value="${objeto.atributo}" />`
- Imagine um objeto Pessoa, com os atributos de nome e idade:

Aluno : `<c:out value="${Pessoa.nome}" />`
com `<c:out value="${Pessoa.idade}" />` anos.

Usando variáveis

- Para setar valores de variáveis:

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

ou

```
<c:set var="nomeDaVariavel">valor</c:set>
```

- Para exibir o valor da variável acima:

```
<c:out value="${nomeDaVariavel}" />
```

Recuperando valores

- Para exibir o valor de parâmetros passados para um formulário :

```
<c:out value="${param.nomeDoParametro}" />
```

- Exemplo:

Bem-vindo <c:out value="\${param.usuario}" />!

Recuperando valores

- Para exibir o valor de parâmetros passados para um formulário via **session**:

```
<c:out value="${sessionScope.nomeDoParametro}" />
```

- Exemplo:

Bem-vindo

```
<c:out value="${sessionScope.usuario}" />!
```


Recuperando valores

- Para exibir o valor de parâmetros passados para um formulário via ***request***:

```
<c:out value="${requestScope.nomeDoParametro}" />
```

- Exemplo:

Bem-vindo

```
<c:out value="${requestScope.usuario}" />!
```



Recuperando valores

Redefina a aplicação sobre Login
para que os dados sejam
recuperados e utilizados na
página **bemVindo.jsp** via
TagLibs

Operadores Condicionais

- O operador condicional para apenas uma opção:

```
<c:if test="${expressão}">
```

...

```
</c:if>
```

- Exemplo:

```
<c:if test="${param.nome == 'nobody'}">
```

Acesso Negado

```
</c:if>
```

Operadores Condicionais

O operador condicional para duas ou mais opções:

<c:choose>

<c:when test="{expressão}">

...

</c:when>

<c:when test="{expressão}">

...

</c:when>

<c:otherwise>

...

</c:otherwise>

</c:choose>

Operadores Condicionais

- Exemplo:

```
<c:choose>
```

```
<c:when test="${hora<12}">
```

Bom dia!

```
</c:when>
```

```
<c:when test="${hora<18}">
```

Boa tarde!

```
</c:when>
```

```
<c:otherwise>
```

Boa noite!

```
</c:otherwise>
```

```
</c:choose>
```



Loops

- O loop **forEach** é usado para percorrer dados de uma coleção ou para iterações simples:

```
<c:forEach var="nome" items="${nomeDaCollection}">  
  <c:out value="${nome}"/>  
</c:forEach>
```

- Ou seja, ele percorrerá a coleção identificada em items, sendo a variável definida nome contendo cada item da coleção, a cada loop realizado.

Loops

- Exemplo:

```
<c:forEach var="aluno" items="${listaAlunos}">  
  <c:out value="${aluno.nome}"/>  
</c:forEach>
```

Loops

- Podemos utiliza-lo também como um FOR simples de controle de iterações:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
    <head>
        <title><c:forEach> Tag Example</title>
    </head>
    <body>
        <c:forEach var="i" begin="1" end="5" step="1">
            Item <c:out value="${i}"/><p>
        </c:forEach>
    </body>
</html>
```




Aplicando *forEach*

Modifique o exercício 2 de fixação de Servlets, utilizando *forEach* para exibição da tabela de parcelamento da página **simulacaoFinaciamento.jsp**, além do uso de tags condicionais para a verificação das linhas pares e ímpares



Formatação e i18n

- Dinamicamente podemos exibir textos de outros idiomas:

```
<fmt:setLocale value="it_IT"/>
```

```
<fmt:setBundle
```

```
  basename="org.apache.taglibs.standard.examples.i  
  18n.Resources" var="itBundle" scope="page"/>
```

```
<fmt:message key="greetingMorning"  
bundle="${itBundle}"/>
```

Resultado: Buon giorno!

Formatação e i18n

- Dinamicamente exibimos facilmente valores numéricos e datas:

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

```
<fmt:formatDate value="${compraDoProduto}"  
type="date" dateStyle="full"/>.
```



Internacionalizando

■ Passos:

- Criar dois arquivos com a extensão **.properties**
>> pacote **internacional**
- Digitar neste arquivos as chaves e os textos:

Sample ResourceBundle properties file

titulo=Internacionalização

ingles=Inglês

portugues=Português

nome=Nome

email=E-mail

enviar=Enviar

Internacionalizando

■ Passos:

- Alternativamente, pode-se adicionar outras localidades >> opções do arquivo de propriedades

Chave	idioma padrão	en_US - English (United States)
titulo	Internacionalização	Internationalization
ingles	Inglês	English
portugues	Português	Portuguese
nome	Nome	Name
email	E-mail	E-mail
enviar	Enviar	Send

- Altere os textos e suas chaves.

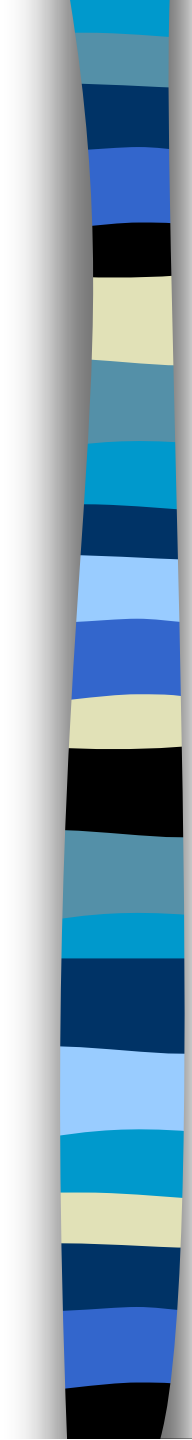
Internacionalizando

- Criar uma página *TesteInternacional.jsp* para teste:

.....

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
<c:choose>
  <c:when test="${param.locale eq 'en_US'}">
    <fmt:setLocale value="en_US" />
  </c:when>
  <c:otherwise>
    <fmt:setLocale value="pt_BR" />
  </c:otherwise>
</c:choose>

<fmt:setBundle basename="br.com.integrator.rotulos"/>
```



```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;
    charset=ISO-8859-1">
    <title><fmt:message key="titulo" /></title>
  </head>
  <body>

    <a href="?locale=en_US"><fmt:message key="ingles" /></a>
    <a href="?locale=pt_BR"><fmt:message key="portugues" /></a>
    <br />
    <form action="">
      <fmt:message key="nome" />: <input type="text"
      name="nome" />
      <br />
      <fmt:message key="email" />: <input type="text"
      name="email" /><br />
      <input type="submit" value="<fmt:message key="enviar" />" />
    </form>
  </body>
</html>
```



Action `<fmt:setLocale>`

- Este action pode ser usada para alterar o local do cliente especificado no processamento de uma página JSP.
- Exemplo:

```
<fmt:setLocale value="en_US" scope="session"/>
```

- **Value:** especifica um código de duas partes que representa o código de idioma ISSO-639 e o código do país ISSO-3166



Exibindo os textos do idioma definido.

- Com local definido, ou pela configuração do browser do cliente ou através de uso da action `<fmt:setLocale />`, o JSTL precisa usar os textos pré-definidos no idioma escolhido para exibir o conteúdo no browser com o idioma especificado por seu local.
- Utiliza-se uma coleção de recursos que é conhecida como ***resource bundle***
 - Utilizada pelo desenvolvedor para cada local que se pretende aderir.
 - É implementada por padrão de uma **chave = valor** em um arquivo de propriedades (extensão .properties).



Actions **<fmt:Bundle />** **<fmt:setBundle />**

- Para habilitar o uso de textos no idioma definido, especificamos o pacote de recursos exigido que forneçam as mensagens localizadas.
- Utiliza-se as tags **<fmt:Bundle />** ou **<fmt:setBundle />** para especificarmos um recurso >> uma vez declarado, pode ser utilizado para fornecer os textos no idioma definido.
- As tags **<fmt:Bundle />** ou **<fmt:setBundle />** embora semelhantes, podem ser usadas de diferentes modos:
- A action **<fmt:Bundle />** é usada para declarar uma localização de contexto l18n para usar por tags dentro de seu corpo:



<fmt:Bundle />

```
<fmt:bundle basename="Rotulos"/>  
    <fmt:message key="rotulos.nome"/>  
    <fmt:message key="rotulos.email"/>  
</fmt:bundle>
```

- O resource bundle com o nome rótulo é declarado para fornecer recursos localizados para as actions <fmt:message />



<fmt:Bundle />

- Como a action **<fmt:bundle />** é projetada para trabalhar com aninhamento da action **<fmt:message />**, um atributo opcional também pode ser usado:

...

<fmt:bundle basename="Rotulos" prefix="rotulos" />

...

- O atributo opcional prefix habilita a colocação de um prefixo pré-definido que é fundamental para qualquer action **<fmt:message />** aninhado tornando seu uso mais simplificado.

<fmt:setBundle />

- Também fornece funcionalidade semelhante a <fmt:bundle />, com uma diferença sutil:
 - Em vez de ter que aninhar qualquer action <fmt:message /> como conteúdo de corpo, a action <fmt:setBundle /> habilita um pacote de recursos a serem armazenados na variável de contexto da aplicação.
 - Assim, qualquer action <fmt:message /> que aparecer, em qualquer parte da página JSP, pode acessar o pacote sem ter que ser declarada continuamente:

```
<fmt:setBundle basename="Rotulos" />
```

```
<fmt:message prefix="rotulos.nome" />
```



Action `<fmt:Message />`

- Usa um parâmetro fundamental, **key**, para extrair a mensagem do pacote de recursos e imprimir com JspWriter.
- Outro parâmetro opcional, **var**, habilita a mensagem localizada a ser armazenada em um parâmetro em vez de ser impressa pelo JspWriter
 - A extensão desta variável pode ser fixada usando o atributo **scope**.