BEA WebLogic Workshop™ Help

Version 8.1 SP4
December 2004

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Table of Contents

JSP Tags Reference

WebLogic Workshop provides JSP tags in the following tag libraries. These JSP tags are designed for use with page flow web projects.

<netui:...> Tags

<netui−data:...> Tags

<netui−template:...> Tags

<netui:...> Tags

Tag	***Summary***
<netui:anchor>	Generates a URL−encoded hyperlink to a specified URI.
<netui:attribute>	Adds an attribute and value to the parent tag.
<netui:base>	Provides the base for every URL on this page.
<netui:bindingUpdateErrors>	A development−time aide that displays messages for data binding update errors that occurred when a form was posted. The messages are displayed on the command window. By default, this tag is disabled when the server is running in Production mode.
<netui:button>	Create a button on your JSP page. Place this tag in a <netui:form> ... </netui:form> tag set.
<netui:checkBox>	Generates a checkbox that binds to a form bean property or databound expression.
<netui:checkBoxGroup>	Groups a collection of CheckBoxOptions, and handles data binding of their values.
<netui:checkBoxOption>	A checkbox whose state is determined by its enclosing CheckBoxGroup.
<netui:content>	Used to display text or the result of an expression to the page.
<netui:error>	Renders an error message with a given error key value.
<netui:errors>	Used to report multiple validation errors.
<netui:exceptions>	Displays formatted exception messages.
<netui:fileUpload>	Uploads a file from the client to the server.
<netui:form>	Represents an input form, associated with a bean whose properties correspond to the various fields of the form.
<netui:formatDate>	A formatter used to format dates.
<netui:formatNumber>	A formatter used to format numbers.

<netui:formatString>	A formatter used to format strings.
<netui:getNetuiTagName>	Returns the value of a specified tagId attribute.
<netui:hidden>	Generates a hidden tag with a given value.
<netui:html>	Generates the html element and performs error handling within its body.
<netui:image>	Places an image file type on your page.
<netui:imageAnchor>	Combines the functionality of the netui:image and netui:anchor tags.
<netui:imageButton>	Combines the functionality of the netui:image and netui:button tags.
<netui:label>	Places formatted or dynamically generated text on the page.
<netui:node>	Instantiates a TreeNode object that will get added to the parent tag (either a Tree or another Node).
<netui:parameter>	Writes a URL parameter to a URL on its parent tag.
<netui:parameterMap>	Provides a read−only data binding expression that points to a map of parameters. Each entry in the map provides a URL parameter that will be added to the parent tag's URL.
<netui:radioButtonGroup>	Defines a group of netui:radioButtonOption elements.
<netui:radioButtonOption>	A radio button whose state is determined by its enclosing netui:RadioButtonGroup.
<netui:rewriteName>	Allows the URL Rewriter to rewrite the name attribute before it is output into the HTML stream
<netui:rewriteURL>	Allows the URL Rewriter to rewrite the url attribute before it is output into the HTML stream
<netui:scriptContainer>	Each scriptContainer will collect all the JavaScript defined for other tags that are embedded with the <netui:scriptContainer> ... </netui:scriptContainer> tag set, and output all the JavaScript in a single <script>. Also each scriptContainer provides optional scoping of its tagID, which allows you to set the scope of the JavaScript's methods.
<netui:select>	Defines a multiple−choice menu or drop−down list within a netui:form element.
<netui:selectOption>	An option whose state is determined by its enclosing netui:selectOption.
<netui:textArea>	Renders a databound TextArea with the given attributes.
<netui:textBox>	Renders a databound TextBox with the given attributes.
<netui:tree>	Renders a tree control represented by a set of TreeNode objects.

<netui−data:...> Tags

Tag	***Summary***

<netui−data:anchorColumn>	Renders an HTML anchor into each data cell in the column.
<netui−data:basicColumn>	Renders data from the data set into the page.
<netui−data:callControl>	Used to call a method on a control.
<netui−data:callMethod>	Used to call a method on an object.
<netui−data:callPageFlow>	Used to call a method on the current page flow controller.
<netui−data:cellRepeater>	A repeating, databound tag that renders its body into each cell of a table of the specified dimensions.
<netui−data:choice>	When nested in a netui−data:repeaterItem tag, allows its body to be rendered conditionally.
<netui−data:choiceMethod>	Used to choose a particular netui−data:choice tag whose body should be rendered for the current data item in the Repeater.
<netui−data:columns>	Provides a container for the columns that will render the header, data, and footer for each column in a netui−data:grid.
<netui−data:declareBundle>	Declare a resource bundle that is available in the bundle data binding context.
<netui−data:declareControl>	Declare a control that is stored in the pageContext attribute map.
<netui−data:declarePageInput>	Lets you specify what data this JSP page expects an action to forward. In the JSP Designer, you can later drag elements of the specified data onto the page's canvas for display.
<netui−data:expressionColumn>	A column that can use data binding expressions in addition to formatters to format the value of a data cell.
<netui−data:getData>	Evaluates an expression and places the result in the JSP's Page Context. Can be used to evaluate objects from forms, page flows, and other objects that can be databound. You can then write a scriplet to access the data by using the getAttribute method of javax.servlet.jsp.PageContext.
<netui−data:grid>	The containing tag of a tag set that renders regular data with the ability to page, sort, and filter the data set.
<netui−data:gridStyle>	Allows parameterization of the style components of the HTML table that a Grid renders.
<netui−data:imageColumn>	In a grid, renders an image into a column.
<netui−data:message>	Allows you to format messages according to any sequence you want, using one or more values from arguments defined in <netui−data:messageArg> tag(s). The results are available to the page context.
<netui−data:messageArg>	Allows you to set values that are used as arguments to the <netui−data:message> tag. The formatted message results are available to the page context.
<netui−data:methodParameter>	Used to add an argument to a method that will be called on some object.

<netui−data:pad>	Affects the number of items that are rendered in a Repeater.
<netui−data:pager>	Allows the grid to render a subset of data on each page and renders the navigation links for moving between grid pages.
<netui−data:repeater>	A markup−generic tag that repeats over a data set, and renders the data onto the page.
<netui−data:repeaterFooter>	Used to render the footer of a Repeater.
<netui−data:repeaterHeader>	Used to render the header of a Repeater.
<netui−data:repeaterItem>	Used to render each item in the data set.

<netui−template:...> Tags

Tag	***Summary***
<netui−template:attribute>	Use the netui−template:attribute element by (1) placing it in a tempate file and (2) setting its value with the netui−template:setAttribute tag. (The netui−template:setAttribute tag is placed on the page that invokes the template file.
<netui−template:includeSection>	Include a netui−template:includeSection element in a template file to mark out content to be used in another JSP page.
<netui−template:section>	Used to mark out content to replace a netui−template:includeSection within a template file.
<netui−template:setAttribute>	Used to set the value of an netui−template:attribute element in a template file.
<netui−template:template>	Used to associate a JSP page with a particular template file.
<netui−template:visible>	Used to turns on or off display of the body content based upon the visible state of the tag.

Related Topics

Guide to Building Page Flows

Using Data Binding in Page Flows

Presenting Complex Data Sets in JSPs

J2EE JSP v1.2 Tags Reference

See the following topics for reference information about J2EE JSP v1.2 tags:

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:plugin Tag

jsp:setProperty Tag

jsp:useBean Tag

Related Topics

netui−tags−html Tags

netui−tags−databinding Tags

netui−tags−template Tags

Tutorial: Your First Page Flow Application

Page Flow and JSP Samples

Getting Started with Page Flows

jsp:fallback Tag

You can use <jsp:fallback> tag as part of the <jsp:plugin> tag. Using <jsp:fallback> in any other context results in a compilation error.

The <jsp:fallback> tag indicates the content to be used by the client browser if the plugin cannot be started, either because OBJECT or EMBED is not supported by the client browser, or due to some other problem. If the plugin can start but the applet or JavaBeans component cannot be found or started, a plugin specific message will be presented. Usually the message is a popup window reporting a ClassNotFoundException.

For more information, see the topic about the jsp:plugin Tag.

Syntax

<jsp:fallback>
  { Text message for user }
</jsp:fallback>

Attributes

None.

Example

```
<jsp:plugin type=applet code="Molecule.class" codebase="/html" >
  <jsp:params>
    <jsp:param name="molecule" value="molecules/benzene.mol"/>
  </jsp:params>
  <jsp:fallback>
    <p> Unable to start plugin </p>
  </jsp:fallback>
</jsp:plugin>
```

Related Topics

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:plugin Tag

jsp:setProperty Tag

jsp:useBean Tag

JSP Tag Reference for Page Flows

jsp:forward Tag

The <jsp:forward> tag forwards the request object containing the client request information from one JSP page to another resource. The target resource can be an HTML file, another JSP page, or a servlet, as long as it is in the same application context as the forwarding JSP page. The lines in the source JSP page after the <jsp:forward> tag are not processed.

You can pass parameter names and values to the target resource by using a <jsp:param> clause. An example of this would be passing the parameter name username (with name="username") and the value nakamura (with value="nakamura") to a servlet as part of the request. If you use <jsp:param>, the target resource should be a dynamic resource that can handle the parameters.

Be careful when using <jsp:forward> with unbuffered output. If you have used the page directive with buffer="none" to specify that the output of your JSP page should not be buffered, and if the JSP page has any data in the out object, using <jsp:forward> will cause an IllegalStateException.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

<jsp:forward page="{relativeURL | <%= expression %>}" />

Or:

```
<jsp:forward page="{relativeURL | <%= expression %>}" >
  <jsp:param name="parameterName"
   value="{parameterValue | <%= expression %>}" /> +
</jsp:forward>
```

Attributes

page="{relativeURL | <%= expression %>}"

The relative URL that locates the resource to be included, or an expression that evaluates to a String equivalent to the relative URL. The relative URL looks like a pathname−−it cannot contain a protocol name, port number, or domain name. The URL can be absolute or relative to the current JSP page. If it is absolute (beginning with a /), the pathname is resolved by your web or application server.

```
<jsp:param name="parameterName"
                value="{parameterValue | <%= expression %>}" />+
```

Sends one or more name/value pairs as parameters to a dynamic resource. The target resource should be dynamic, that is, a JSP page, servlet, or other resource that can process the data that is sent to it as parameters. You can use more than one <jsp:param> clause if you need to send more than one parameter to the target resource. The name attribute specifies the parameter name and takes a case−sensitive literal string as a value. The value attribute specifies the parameter value and takes either a case−sensitive literal string or an expression that is evaluated at request time.

Examples

```
<jsp:forward page="/servlet/login" />
<jsp:forward page="/servlet/login">
   <jsp:param name="username" value="nakamura" />
</jsp:forward>
```

Related Topics

jsp:fallback Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:plugin Tag

jsp:setProperty Tag

jsp:useBean Tag

JSP Tag Reference for Page Flows

jsp:getProperty Tag

The <jsp:getProperty> tag gets a bean property value using the property's getter methods and inserts the value into the response. You must create or locate a bean with <jsp:useBean> before you use <jsp:getProperty>. The <jsp:getProperty> tag has a few limitations:

- You cannot use <jsp:getProperty> to retrieve the values of an indexed property.
- You can use <jsp:getProperty> with JavaBeans components, but not with Enterprise JavaBeans. As alternatives, you can write a JSP page that retrieves values from a bean that in turn retrieves values from an enterprise bean, or you can write a custom tag that retrieves values from an enterprise bean directly.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

<jsp:getProperty name="beanInstanceName" property="propertyName" />

Attributes

name="beanInstanceName"

The name of an object (usually an instance of a bean) as declared in a jsp:usebean tag.

property="propertyName"

The name of the bean property whose value you want to display. The property is declared as a variable in a bean and must have a corresponding getter method (for more information on declaring variables and writing getter methods in beans, see http://java.sun.com/products/javabeans/docs/).

Examples

```
<jsp:useBean id="calendar" scope="page" class="employee.Calendar" />
<h2>
Calendar of <jsp:getProperty name="calendar" property="username" />
</h2>
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:include Tag

The <jsp:include> tag lets you include either a static or dynamic resource in a JSP page. The results of including static and dynamic resources are quite different. If the resource is static, its content is included in the calling JSP page. If the resource is dynamic, it acts on a request and sends back a result that is included in the JSP page. When the include action is finished, the JSP container continues processing the remainder of the JSP page.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

<jsp:include page="{ relativeURL | <%= expression %> }" flush="true | false" />

Or:

```
<jsp:include page="{relativeURL | <%= expression %>}" flush="true| false" >
  <jsp:param name="parameterName"
  value="{parameterValue | <%= expression %>}" />
</jsp:include>
```

Attributes

page="{ relativeURL | <%= expression %> }"

The relative URL that locates the resource to be included, or an expression that evaluates to a String equivalent to the relative URL. The relative URL looks like a pathname−−it cannot contain a protocol name, port number, or domain name. The URL can be absolute or relative to the current JSP page. If it is absolute (beginning with a /), the pathname is resolved by your web or application server.

flush="true | false"

If the page output is buffered and the flush attribute is given a true value, the buffer is flushed prior to the inclusion, otherwise the buffer is not flushed. The default value for the flush attribute is false.

```
<jsp:param name="parameterName"
                value="{parameterValue | <%= expression %>}" />
```

The <jsp:param> clause allows you to pass one or more name/value pairs as parameters to an included resource. The included resource should be dynamic, that is, a JSP page, servlet, or other resource that can process the parameter. You can use more than one <jsp:param> clause if you want to send more than one parameter to the included resource. The name attribute specifies the parameter name and takes a case−sensitive literal string. The value attribute specifies the parameter value and takes either a case−sensitive literal string or an expression that is evaluated at request time.

Examples

```
<jsp:include page="scripts/login.jsp" />
<jsp:include page="copyright.html" />
<jsp:include page="/index.html" />
<jsp:include page="scripts/login.jsp">
   <jsp:param name="username" value="jsmith" />
</jsp:include>
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:param Tag

jsp:params Tag

jsp:plugin Tag

jsp:setProperty Tag

jsp:useBean Tag

JSP Tag Reference for Page Flows

jsp:param Tag

The <jsp:param> tag is used to provide key/value information. This tag is used in the jsp:include, jsp:forward and jsp:params tags. A translation error
occurs if the element is used elsewhere. When executing jsp:include or jsp:forward, the included page or forwarded page sees the original request object, with the original parameters augmented with the new parameters, with new values taking precedence over existing values when applicable. The scope of the new parameters is the jsp:include or jsp:forward call; that is, in the case of an jsp:include the new parameters (and values) will not apply after the include. For example, if the request has a parameter A=foo and a parameter A=bar is specified for forward, the forwarded request has A=bar,foo. Note that the
new param has precedence.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

<jsp:param name="parameterName"
value="{parameterValue | <%= expression %>}" />

Attributes

name="parameterName"

The name of the parameter.

value="{parameterValue | <%= expression %>}" />+

Specifies the parameter value and takes either a case−sensitive literal string or an expression that is evaluated at request time.

Examples

```
<jsp:param page="scripts/login.jsp" />
<jsp:param page="copyright.html" />
<jsp:param page="/index.html" />
<jsp:param page="scripts/login.jsp">
   <jsp:param name="username" value="jsmith" />
</jsp:param>
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:params Tag

The jsp:params tag is part of the jsp:plugin tag and can only occur as a direct child of a <jsp:plugin> tag. Using the jsp:params tag in any other
context results in a translation−time error.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

```
<jsp:params>
    [ <jsp:param name="parameterName"
        value="{parameterValue | <%= expression %>}" /> ]
</jsp:params>
```

Attributes

```
<jsp:param name="parameterName"
                value="{parameterValue | <%= expression %>}" />+
```

The <jsp:param> clause allows you to pass one or more name/value pairs as parameters to an included resource. The included resource should be dynamic; that is, a JSP page, servlet, or other resource that can process the parameter. You can use more than one <jsp:param> clause if you want to send more than one parameter to the included resource. The name attribute specifies the parameter name and takes a case−sensitive literal string. The value attribute specifies the parameter value and takes either a case−sensitive literal string or an expression that is evaluated at request time.

Examples

```
<jsp:plugin type=applet code=”Molecule.class” codebase=”/html” >
  <jsp:params>
    <jsp:param name=”molecule” value=”molecules/benzene.mol”/>
  </jsp:params>
  <jsp:fallback>
    <p> Unable to start plugin </p>
  </jsp:fallback>
</jsp:plugin>
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:plugin Tag

jsp:setProperty Tag

jsp:useBean Tag

JSP Tag Reference for Page Flows

jsp:plugin Tag

The <jsp:plugin> tag enables a JSP page author to generate HTML that contains the appropriate client browser dependent constructs (OBJECT or EMBED), which will result in the download of the Java Plugin software (if required) and subsequent execution of the specified applet or JavaBeans component.

In the rendered JSP, the <jsp:plugin> tag is replaced by either an <object> or <embed> tag, as appropriate for the requesting user agent, and emitted into the output stream of the response. The attributes of the <jsp:plugin> tag provide configuration data for the presentation of the tag.

The <jsp:params> tag indicate the parameters to the applet or JavaBeans component.

The <jsp:fallback> tag indicates the content to be used by the client browser if the plugin cannot be started, either because OBJECT or EMBED is not supported by the client browser, or due to some other problem. If the plugin can
start but the applet or JavaBeans component cannot be found or started, a plugin specific message is presented to the user. Usually the message is in a popup window reporting a ClassNotFoundException.

You are not required to bundle the actual plugin code with the JSP container. You may use a reference to the plugin location.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

```
<jsp:plugin
  type="bean|applet"
  code="classFileName"
  codebase="classFileDirectoryName"
  [ name="instanceName" ]
  [ archive="URIToArchive, ..." ]
  [ align="bottom|top|middle|left|right" ]
  [ height="{displayPixels | <%= expression %>}"]
  [ width="{displayPixels | <%= expression %>}"]
  [ hspace="leftRightPixels" ]
  [ vspace="topBottomPixels" ]
  [ jreversion="JREVersionNumber | 1.2" ]
  [ nspluginurl="URLToPlugin" ]
  [ iepluginurl="URLToPlugin" ] >
  [ <jsp:params>
    [ <jsp:param name="parameterName"
    value="{parameterValue | <%= expression %>}" /> ]+
  </jsp:params> ]
  [ <jsp:fallback> text message for user </jsp:fallback> ]
</jsp:plugin>
```

Attributes

type="bean | applet"

The type of object the plug−in will execute. You must specify either bean or applet, as this attribute has no default value.

code="classFileName"

The name of the Java class file the plug−in will execute. You must include the .class extension in the name. The class file you specify should be in the directory named in the codebase attribute.

codebase="classFileDirectoryName"

The directory (or path to the directory) that contains the Java class file the plug−in will execute. If you do not supply a value, the path of the JSP page that calls <jsp:plugin> is used.

name="instanceName"

A name for the instance of the bean or applet, which makes it possible for applets or Beans called by the same JSP page to communicate with each other.

archive="URIToArchive, ..."

A comma−separated list of pathnames that locate archive files that will be preloaded with a class loader located in the directory named in codebase. The archive files are loaded securely, often over a network, and typically improve the applet's performance.

align="bottom | top | middle | left | right"

The position of the image, object, or applet. The position descriptions listed below use the term text line to mean the line in the viewable JSP page that corresponds to the line in the JSP page where the <jsp:plugin> tag appears. The allowed values for align are as follows:

bottom: Aligns the bottom of the image with the baseline of the text line.

top: Aligns the top of the image with the top of the text line.

middle: Aligns the vertical center of the image with the baseline of the text line.

left: Floats the image to the left margin and flows text along the image's right side.

right: Floats the image to the right margin and flows text along the image's left side.

height="{displayPixels | <%= expression %>}"
width="{displayPixels | <%= expression %>}"

The initial height and width, in pixels, of the image the applet or bean displays, not counting any windows or dialog boxes the applet or bean brings up.

hspace="leftRightPixels"
vspace="topBottomPixels"

The amount of space, in pixels, to the left and right (or top and bottom) of the image the applet or bean displays. The value must be a nonzero number. Note that hspace creates space to both the left and right and vspace creates space to both the top and bottom.

jreversion="JREVersionNumber | 1.2"

The version of the Java Runtime Environment (JRE) the applet or bean requires. The default value is 1.2.

nspluginurl="URLToPlugin"

The URL where the user can download the JRE plug−in for Netscape Navigator. The value is a full URL, with a protocol name, optional port number, and domain name.

iepluginurl="URLToPlugin"

The URL where the user can download the JRE plug−in for Internet Explorer. The value is a full URL, with a protocol name, optional port number, and domain name.

<jsp:params>
    [<jsp:param name="parameterName"
        value="{parameterValue | <%= expression %>}" />]
</jsp:params>

The parameters and values that you want to pass to the applet or bean. To specify more than one parameter value, you can use more than one <jsp:param> tag within the <jsp:params> tag. The name attribute specifies the parameter name and takes a case−sensitive literal string. The value attribute specifies the parameter value and takes either a case−sensitive literal string or an expression that is evaluated at runtime. If the dynamic resource you are passing the parameter to is an applet, it reads the parameter with the java.applet.Applet.getParameter method.

<jsp:fallback> { Text message for user } </jsp:fallback>

A text message to display for the user if the plug−in cannot be started. If the plug−in starts but the applet or bean does not, the plug−in usually displays a popup window explaining the error to the user.

Example

```
<jsp:plugin type=applet code=”Molecule.class” codebase=”/html” >
  <jsp:params>
    <jsp:param name=”molecule” value=”molecules/benzene.mol”/>
  </jsp:params>
  <jsp:fallback>
    <p> Unable to start plugin </p>
  </jsp:fallback>
</jsp:plugin>
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:setProperty Tag

jsp:useBean Tag

JSP Tag Reference for Page Flows

jsp:setProperty Tag

The <jsp:setProperty> tag sets the value of one or more properties in a bean, using the bean's setter methods. You must declare the bean with <jsp:useBean> before you set a property value with <jsp:setProperty>. Because <jsp:useBean> and <jsp:setProperty> work together, the bean instance names they use must match. That is, the value of name in <jsp:setProperty> and the value of id in <jsp:useBean> must be the same.

You can use <jsp:setProperty> to set property values in several ways:

- By passing all of the values the user enters (stored as parameters in the request object) to matching properties in the bean
- By passing a specific value the user enters to a specific property in the bean
- By setting a bean property to a value you specify as either a String or an expression that is evaluated at runtime

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

```
<jsp:setProperty name="beanInstanceName"
{
  property="*" |
  property="propertyName" [ param="parameterName" ] |
  property="propertyName" value="{stringLiteral| <%= expression %>}"
}
/>
```

Attributes

name="beanInstanceName"

The name of an instance of a bean that has already been created or located with a <jsp:useBean> tag. The value of name must match the value of id in <jsp:useBean>. The <jsp:useBean> tag must appear before <jsp:setProperty> in the JSP page.

property="*"

Stores all of the values of request parameters in bean properties. The names of the bean properties must match the names of the request parameters. A bean property is usually defined by a variable declaration with matching getter and setter methods. When you use property="*", the bean properties are not necessarily set in the order in which they appear in the HTML form or the bean. If the order in which the properties are set is important to how your bean works, use the syntax form property="propertyName" [param="parameterName"]. Or rewrite your bean so that the order of setting properties is not important.

property="propertyName" [param="parameterName"]

Sets one bean property to the value of one request parameter. In the syntax, property specifies the name of the bean property and param specifies the name of the request parameter by which data is being sent from the

client to the server. If the bean property and the request parameter have different names, you must specify both property and param. If they have the same name, you can specify property and omit param. If a parameter has an empty or null value, the corresponding bean property is not set.

property="propertyName" value="{ String | <%= expression %>}"

Sets one bean property to a specific value. The value can be a String or an expression that is evaluated at runtime. If the value is a String, it is converted to the bean property's data type according to the conversion rules shown above in TABLE 1. If it is an expression, its value must have a data type that matches the the data type of the value of the expression must match the data type of the bean property. If the parameter has an empty or null value, the corresponding bean property is not set. You cannot use both the param and value attributes in a <jsp:setProperty> tag.

Examples

```
<jsp:setProperty name="mybean" property="*" />
<jsp:setProperty name="mybean" property="username" />
<jsp:setProperty name="mybean" property="username" value="Steve" />
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:plugin Tag

jsp:useBean Tag

JSP Tag Reference for Page Flows

jsp:useBean Tag

The <jsp:useBean> tag locates or instantiates a JavaBeans component. The <jsp:useBean> tag first attempts to locate an instance of the bean. If the bean does not exist, <jsp:useBean> instantiates it from a class or serialized template. To locate or instantiate the bean, <jsp:useBean> takes the following steps, in this order:

1. Attempts to locate a bean with the scope and name you specify.
2. Defines an object reference variable with the name you specify.
3. If it finds the bean, stores a reference to it in the variable. If you specified type, gives the bean that type.
4. If it does not find the bean, instantiates it from the class you specify, storing a reference to it in the new variable. If the class name represents a serialized template, the bean is instantiated by java.beans.Beans.instantiate.
5. If <jsp:useBean> has instantiated (rather than located) the bean, and if it has body tags or JSP tags (between <jsp:useBean> and </jsp:useBean>), executes the body tags.

For more information, see the JavaServer Pages (JSP) v1.2 Syntax Reference on the Sun Microsystems® web site.

Syntax

```
<jsp:useBean id="beanInstanceName" scope="page|request|session|application"
{
  class="package.class" [ type="package.class" ]|
  beanName="{package.class |
  <%= expression %>}" type="package.class" |
  type="package.class"
}
{ /> | > other tags </jsp:useBean> }
```

Attributes

id="beanInstanceName"

A variable that identifies the bean in the scope you specify. You can use the variable name in expressions or scriptlets in the JSP page. The name is case sensitive and must conform to the naming conventions of the scripting language used in the JSP page. If you use the Java programming language, the name must conform to the naming conventions in the Java Language Specification. If the bean has already been created by another <jsp:useBean> tag, the value of id must match the value of id used in the original <jsp:useBean> tag.

scope="page|request|session|application"

The scope in which the bean exists and the variable named in id is available. The default value is page. The meanings of the different scopes are as follows:

page: You can use the bean within the JSP page with the <jsp:useBean> tag or any of the page's static include files, until the page sends a response back to the client or forwards a request to another resource.

request: You can use the bean from any JSP page processing the same request, until a JSP page sends a response to the client or forwards the request to another resource. You can use the request object to access the bean, for example, request.getAttribute(beanInstanceName).

session: You can use the bean from any JSP page in the same session as the JSP page that created the bean. The bean exists across the entire session, and any page that participates in the session can use it. The page in which you create the bean must have a page directive with session="true".

application: You can use the bean from any JSP page in the same application as the JSP page that created the bean. The bean exists across an entire JSP application, and any page in the application can use the bean.

class="package.class"

Instantiates a bean from a class, using the new keyword and the class constructor. The class must not be abstract and must have a public, no−argument constructor. The package and class name are case sensitive.

type="package.class"

If the bean already exists in the scope, gives the bean a data type other than the class from which it was instantiated. The value of type must be a superclass of class or an interface implemented by class. If you use type without class or beanName, no bean is instantiated. The package and class name are case sensitive.

class="package.class" type="package.class"

Instantiates a bean from the class named in class and assigns the bean the data type you specify in type. The value of type can be the same as class, a superclass of class, or an interface implemented by class. The class you specify in class must not be abstract and must have a public, no−argument constructor. The package and class names you use with both class and type are case sensitive.

beanName="{package.class | <%= expression %>}" type="package.class"

Instantiates a bean from a class, a serialized template, or an expression that evaluates to a class or serialized template. When you use beanName, the bean is instantiated by the java.beans.Beans.instantiate method. The Beans.instantiate method checks whether the package and class you specify represents a class or a serialized template. If they represent a serialized template, Beans.instantiate reads the serialized form (which has a name like package.class.ser) using a class loader.

Examples

```
<jsp:useBean id="cart" scope="session" class="session.Carts" />
<jsp:setProperty name="cart" property="*" />
<jsp:useBean id="checking" scope="session" class="bank.Checking" >
   <jsp:setProperty name="checking" property="balance" value="0.0" />
</jsp:useBean>
```

Related Topics

jsp:fallback Tag

jsp:forward Tag

jsp:getProperty Tag

jsp:include Tag

jsp:param Tag

jsp:params Tag

jsp:plugin Tag

jsp:setProperty Tag

JSP Tag Reference for Page Flows