BEA WebLogic Portal

JavaServer Page Guide

Release 7.0 Service Pack 5
Document Date: April 2004

Copyright

Restricted Rights Legend

Trademarks or Service Marks

WebLogic Portal JavaServer Page Guide

Part Number	Date	Software Version
N/A	April 2004	7.0 Service Pack 5

Contents

1. Introduction

2. Portal and Portlet Management JSP Tags

3. Personalization JSP Tags

4. Navigation (Webflow) JSP Tags

5. Catalog Development JSP Tags

6. Event and Behavior Tracking JSP Tags

A. JSP Templates

B. Tuning JSP Performance

C. JSP Tag Reference (by Name)

Preface

Welcome to the WebLogic Portal JavaServer Page Guide tour. In addtion to this document, we encourage you to use the following resources, as well.

Finding documentation online BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the “e-docs” Product Documentation page at **http://e-docs.bea.com**.

Providing documentation feedback Your feedback on the BEA WebLogic Portal documentation is important to us. Send us e-mail at **docsupport@bea.com** if you have questions or comments. Please indicate that you are using the documentation for the WebLogic Portal version **Product Version:** .

Contacting BEA WebSUPPORT If you have any questions about this version of WebLogic Portal, or if you have problems installing and running WebLogic Portal, contact BEA Customer Support through BEA WebSUPPORT at **http://www.bea.com** or by using the contact information provided on the Customer Support Card in the product package.

1 Introduction

Welcome to the WebLogic Portal JavaServer Page Guide. This document supplements the *Development Guide* by providing detailed descriptions of the JSP tags used to develop portals and other JSP features available with WebLogic Portal. It contains information on the following subjects:

- Tag List by Task
- Tag List By Name
- JSP Template Information

Tag List by Task

This document is organized by the various tasks you might need to accomplish while developing portals and portlets; therefore, a single chapter might document tags from multiple tag libraries. Table 1-1 lists the tag libraries by the chapter and task in which their tags appear.

Table 1-1 JSP Tag Libraries

Tag Library	Used in Task	Chapter
ad.tld	Ad placeholders	Personalization JSP Tags
catalog.tld	Catalog Service Management	Catalog Development JSP Tags
cm.tld	Content Management	Personalization JSP Tags
eb.tld	E-business Service Management	Catalog Development JSP Tags

Table 1-1 JSP Tag Libraries

Tag Library	Used in Task	Chapter
es.tld	Personaliation Utilities	Personalization JSP Tags
i18n.tld	Internationalization Management	Personalization JSP Tags
ph.tld	Placeholder Management	Personalization JSP Tags
portal.tld	Portal Management	Portal and Portlet Management JSP Tags
portlet.tld	Portlet Management	Portal and Portlet Management JSP Tags
productTracking.tld	Event and Behavior Tracking Management	Event and Behavior Tracking JSP Tags
ps.tld	Property Set Management	Personalization JSP Tags
pz.tld	Personalization Management	Personalization JSP Tags
tracking.tld	Event and Behavior Tracking Management	Event and Behavior Tracking JSP Tags
um.tld	User Management	Personalization JSP Tags
util.tld	Portal/Portlet Management	Portal and Portlet Management JSP Tags
webflow.tld	Navigation Management	Navigation (Webflow) JSP Tags
wl	WebLogic Utilities	Personalization JSP Tags

Tag List By Name

In addition to the task-oriented tag descriptions, JSP Tag Reference (by Name) lists all tags in alphabetical order with no regard to the tasks supported. This appendix includes a description of the tag and identifies its tag library. Also, if you are reading this document online (for example, in a Web browser or a PDF viewer), by clicking the tag name, you can see the detailed description of the tag as provided in the task-oriented chapter.

JSP Template Information

This document also contains two appendices, JSP Templates and Tuning JSP Performance, that apply to BEA-supplied JSPs.

- JSP Templates lists and describes all of the JSP templates available with your installation of WebLogic Portal.

Tuning JSP Performance contains instructions for improving JSP performance by tuning its compile and update phases to optimize those capabilities.

2 Portal and Portlet Management JSP Tags

WebLogic Portal includes a set of JSP tags designed to facilitate the development of portals and portlets. Use these predefined tags to reduce the amount of Java code required in your JSP page. This topic explains how to import each set of tags into your Web pages, and describes the purpose of each tag.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

This topic includes the following sections:

- Portlet Tag Libraries
- Portal Tag Libraries
- Utility Tag Libraries

Portlet Tag Libraries

The following section describes the portlet tags and their attributes.

<portlet:createWebflowURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreatePortletURLTag CreateWebflowURLTagExtraInfo

The `<portlet:createWebflowURL>` tag is used in a portlet to dynamically create a Webflow URL in a JSP. The Webflow URL may include the protocol, domain name, port, Web application URI, `WebflowServlet` URI, and query string.

Table 2-1 describes the `<portlet:createWebflowURL>` tag attributes.

Table 2-1 `<portlet:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	The page from which the event is generated.	R
portletName	No	String	The relevant portlet ID. This is usually automatically generated, but can be modified.	R
portletNamespace	No	String	Webflow namespace in which the event will be invoked. This does not need to be the default Webflow namespace for the portlet.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
event	Yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R

Table 2-1 `<portlet:createWebflowURL>` **Tag Attributes**

Tag Attribute	Required	Type	Description	R/C
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	C

Table 2-1 `<portlet:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				C

Table 2-1 `<portlet:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-1 `<portlet:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	C

Example

Listing 2-1 illustrates how to use the `<portlet:createWebflowURL>` JSP tag.

Listing 2-1 Using <portlet:createWebflow>

```
<%@ taglib uri="portlet.tld" prefix="portlet" %>...

<center>
<font size="6" color="green">Portlet 2 - Page 1</font><BR>
<p>
<p>
Portlet Webflow Test:
<p>
<a href="<portlet:createWebflowURL event="switch"/>">Next Page</a>
<p>
</center>
```

<portlet:form>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	PortletFormTag

The `<portlet:form>` tag is used in a JSP to dynamically generate an HTML form tag. This tag is not as sophisticated as the `<portlet:validatedForm>` tag, but is simpler. See <portlet:validatedForm> tag for more information.

Table 2-2 describes the `<portlet:form>` tag attributes.

Table 2-2 `<portlet:form>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	The page from which the event is generated.	R

Table 2-2 `<portlet:form>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
portletName	No	String	The relevant portlet ID. This is usually automatically generated, but can be modified.	R
portletNamespace	No	String	Webflow namespace in which the event will be invoked. This does not need to be the default Webflow namespace for the portlet.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
event	yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R

Table 2-2 `<portlet:form>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				C

Table 2-2 `<portlet:form>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-2 `<portlet:form>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information..	C
hide	No	String	If set to `false`, the namespace, origin, and event will be displayed on the command line instead of as hidden form fields. Valid values are `true` and `false`. The default value is `true`.	C
method	No	String	The method to be used for the form. Valid values are `get` and `post`. The default value is `post`.	C
name	No	String	The name of the form.	C

Table 2-2 <portlet:form> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
htmlAttributes	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R

<portlet:validatedForm>

Tag Library	portlet.tld
Import Statement	<%@ taglib uri="portlet.tld" prefix= "portlet" %>
Classes Implemented	PortletValidatedFormTag WebflowValidatedFormTagExtraInfo

The <portlet:validatedForm> tag is used to dynamically generate HTML forms that can be validated. When a Web site visitor enters invalid information, the visitor's input is preserved and redisplayed with an appropriate error message.

Table 2-3 describes the <portlet:ValidatedForm> tag attributes.

Table 2-3 <portlet:validatedForm> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	The page from which the validated form URL is generated	R
portletName	No	String	The relevant portlet ID. This is usually automatically generated, but can be modified.	R
portletNamespace	No	String	Webflow namespace in which the event will be invoked. This does not need to be the default Webflow namespace for the portlet.	R

Table 2-3 <portlet:validatedForm> (Continued)Tag Attributes

Tag Attribute	Required	Type	Description	R/C
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
event	yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	C

Table 2-3 <portlet:validatedForm> (Continued)Tag Attributes

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VAL`
			`UE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				C

Table 2-3 <portlet:validatedForm> (Continued)Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-3 <portlet:validatedForm> (Continued)Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	C
hide	No	String	If set to `false`, the namespace, origin, and event will be displayed on the command line instead of as hidden form fields. Valid values are `true` and `false`. The default value is `true`.	C
method	No	String	The method to be used for the form. Valid values are `get` and `post`. The default value is `post`.	C
name	No	String	The name of the form.	C

Table 2-3 <portlet:validatedForm> (Continued)Tag Attributes

Tag Attribute	Required	Type	Description	R/C
validStyle	No	String	The style used to format the HTML field when it is valid.	R
invalidStyle	No	String	The style used to format the HTML field or the message when the field is invalid.	R
unspecifiedStyle	No	String	Used to specify the initial style of the HTML field before validation occurs.	R
styleId	No	String	Scripting variable that will be set to one of invalidStyle, unSpecifiedStyle, or validStyle, depending on the field's status: `valid`, `invalid`, `unspecified`. Can be used for finer control of formatting the HTML form.	R
applyStyle	No	String	Applies the associated style as indicated by the field status to the message, the field, or to none. Therefore, valid values are `message`, `field`, and `none`. The default value is `message`.	C
messageAlign	No	String	Indicates whether to align the error message above the field, to the right of the field, or below the field. Therefore, value values are `top`, `right`, and `bottom`. The default value is `right`.	C
htmlAttributes	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R

Example

The following code sample illustrates how to use the `<portlet:validatedForm>` JSP tag:

Listing 2-2 Using `<portlet:validatedForm>`

```
<%@ taglib uri="portlet.tld" prefix="portlet" %>
<%@ taglib uri="webflow.tld" prefix="webflow" %>


<center>
<% String validStyle="color: black; font-family: Arial"; %>
<% String invalidStyle="color: darkred; font-style: Arial"; %>
<%-- If there was an InvalidFormDataException thrown display the
message --%>
<font size="3" color="darkred"><portlet:getException/></font>
<br>
<webflow:getProperty id="welcomeStr" property="WELCOME_STRING"
type="java.lang.String" scope="request" namespace="portlet3"/>
<% if (welcomeStr == null || welcomeStr.length() < 1)
   {
%>
     <portlet:validatedForm event="button.go" applyStyle="message"
messageAlign="right"
       validStyle="<%=validStyle%>"
invalidStyle="<%=invalidStyle%>"
unspecifiedStyle="<%=validStyle%>" >
           <p>
            Welcome :
           <webflow:text name="welcome" value="Hello" size="15"
maxlength="30" /> <br>
           <input type="submit"  name="Submit"/>
</portlet:validatedForm>
     <br>
<%
   }
   else
   {
%>
       <font size="5" color="green"><%= welcomeStr %></font>
<%
   }
%>
</center>
<p>
```

<portlet:createPortletEditURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreateEditURLTag

The `<portlet:createPortletEditURL>` tag generates a webflow URL that represents editing a portlet.

Table 2-4 describes the `<portlet:createPortletEditURL>` tag attributes.

Table 2-4 <portlet:createPortletEditURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	`The page from which the edit URL is generated.`	R
portletName	No	String	The relevant portlet ID. This is usually automatically generated, but can be modified.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R

Table 2-4 <portlet:createPortletEditURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-4 <portlet:createPortletEditURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE`
			`</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-4 <portlet:createPortletEditURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-4 <portlet:createPortletEditURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information..	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Example

Listing 2-3 illustrates how to use the `<portlet:createPortletEditURL>` JSP tag:

Listing 2-3 Using `<portlet:createPortletEditURL>`

```
        }
        if (PortletRenderHelper.isEditable(request))
        {
%>
             <td width="1%" valign="middle"><a
href="<portlet:createPortletEditURL httpsInd='http'/>">
      <img src="<webflow:createResourceURL
```

```
resource='<%=imagesPath+"portlet_edit.gif"%>'/>" hspace="1"
vspace="2" border="0" alt="Edit" align="absmiddle"></a></td>
<%
```

<portlet:createPortletUneditURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreatePortletUneditURLTag

The `<createPortletUneditURL>` JSP tag generates a webflow URL that represents leaving the edit URL and moving to another page.

Table 2-5 describes the `<portlet:createPortletUneditURL>` tag attributes.

Table 2-5 `<portlet:createPortletUneditURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	The page from which the unedit URL is generated.	R
portletName	No	String	The portlet ID. This is usually automatically generated, but can be modified.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R

Table 2-5 `<portlet:createPortletUneditURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-5 <portlet:createPortletUneditURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are calculate, http, and https.
			Calculate will yield HTTPS if any node in the origin/event branch chain list is matched under the HTTPS_URL_PATTERNS context parameter in the application's WEB-INF/web.xml file. Calculate is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for HTTPSIND_DEFAULT_VALUE in web.xml is used. If no value is found there, calculate is used.
			If an invalid value is used for this attribute, calculate is used.
			The following example shows a protocol set in web.xml. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALU`
			`E</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of HTTP, HTTPS or CALCULATE can be specified in all uppercase or all lowercase.
				R

Table 2-5 `<portlet:createPortletUneditURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-5 `<portlet:createPortletUneditURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

<portlet:createPortletMinimizeURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreatePortletMinimizeURLTag

The `<portlet:createPortletMinimizeURL>` JSP tag generates a webflow URL that represents minimizing a portlet.

Table 2-6 describes the `<portlet:createPortletMinimizeURL>` tag attributes.

Table 2-6 `<portlet:createPortletMinimizeURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	`The page from which the minimize URL is generated.`	R
portletName	No	String	The portlet ID. This is usually automatically generated, but can be modified.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R

Table 2-6 `<portlet:createPortletMinimizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALU`
			`E</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-6 `<portlet:createPortletMinimizeURL>` **Tag Attributes (Continued)**

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-6 `<portlet:createPortletMinimizeURL>` **Tag Attributes (Continued)**

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Example

Listing 2-4 illustrates how to use the `<portlet:createPortletMinimizeURL>` JSP tag:

Listing 2-4 Using `<portlet:createPortletMinimizeURL>`

```
        }
        if (PortletRenderHelper.isMinimizable(request))
        {
%>

    <td width="1%" valign="middle"><a
href="<portlet:createPortletMinimizeURL httpsInd='http'/>">
      <img src="<webflow:createResourceURL
resource='<%=imagesPath+"portlet_min.gif"%>'/>" hspace="1"
vspace="2" border="0" alt="Minimize" align="absmiddle"></a></td>
<%
```

<portlet:createPortletUnminimizeURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreateUnminimizeURLTag

The `<portlet:createPortletUnminimizeURL>` tag generates a webflow URL that represents unminimizing a portlet.

Table 2-7 describes the `<portlet:createPortletUnminimizeURL>` tag attributes.

Table 2-7 `<portlet:createPortletUnminimizeURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	The page from which the unminimize URL is generated.	R
portletName	No	String	The portlet ID. This is usually automatically generated, but can be modified.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-7 `<portlet:createPortletUnminimizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-7 `<portlet:createPortletUnminimizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-7 `<portlet:createPortletUnminimizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information..	R

Example

Listing 2-5 illustrates how to use the `<portlet:createPortletUnminimizeURL>` JSP tag:

Listing 2-5 Using `<portlet:createPortletUnminimizeURL>`

```
<%--  Create a link to unminimize the portlet --%>
<td width="1%" valign="middle">
   <a href="<portlet:createPortletUnminimizeURL/>">
   <img src="<webflow:createResourceURL resource=
   '<%=imagesPath+"portlet_unmin.gif"%>'/>" hspace="1" vspace="2"
   border="0" alt="Restore" align="absmiddle"></a></td>
```

<portlet:createPortletMaximizeURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreateMaximizeURLTag

The `<portlet:createPortletMaximizeURL>` JSP tag generates a webflow URL that represents maximizing a portlet.

Table 2-8 describes the `<portlet:createPortletMaximizeURL>` tag attributes.

Table 2-8 `<portlet:createPortletMaximizeURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	`The page from which the maximize URL is generated.`	R
portletName	No	String	The portlet ID. This is usually automatically generated, but can be modified.	R

Table 2-8 `<portlet:createPortletMaximizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-8 `<portlet:createPortletMaximizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALU`
			`E</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-8 `<portlet:createPortletMaximizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-8 `<portlet:createPortletMaximizeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information..	R

Example

Listing 2-6 illustrates how to use the `<portlet:createPortletMaximizeURL>` JSP tag:

Listing 2-6 Using `<portlet:createPortletMaximizeURL>`

```
<%
        }
        if (PortletRenderHelper.isMaximizable(request))
        {
%>

    <td width="1%" valign="middle">
      <%-- Create the link to maximize the portlet --%>
      <a href="<portlet:createPortletMaximizeURL
         httpsInd='http'/>"><img src=
         "<webflow:createResourceURL
         resource='<%=imagesPath+"portlet_max.
         gif"%>'/>" hspace="1" vspace="2" border="0"
         alt="Maximize" align="absmiddle"></a></td>
<%
```

<portlet:createPortletUnmaximizeURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreateUnmaximizeURLTag

The `<portlet:createPortletUnmaximizeURL>` JSP Tag generates a webflow URL that represents unmaximizing a portlet.

Table 2-8 describes the `<portlet:createPortletUnmaximizeURL>` tag attributes.

Table 2-9 <portlet:createPortletUnmaximizeURL>

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	The page from which the unmaximize URL is generated.	R
portletName	No	String	The portlet ID. This is usually automatically generated, but can be modified.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-9 <portlet:createPortletUnmaximizeURL> (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALU`
			`E</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-9 <portlet:createPortletUnmaximizeURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-9 <portlet:createPortletUnmaximizeURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Example

The following code sample illustrates how to use the `<portal:createPortletUnamximizeURL>` JSP tag:

Listing 2-7 Using `<portlet:createPortletUnmaximizeURL>`

```
<%--  Create a link to unmaximize the portlet --%>
<td width="1%" valign="middle">
  <a href="<portlet:createPortletUnmaximizeURL/>">
     <img src="<webflow:createResourceURL
     resource='<%=imagesPath+"portlet_unmax.
     gif"%>'/>" hspace="1" vspace="2" border="0"
     alt="Unmaximize"></a></td>
```

<portlet:createPortletFloatURL>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	CreateFloatURLTag

The `<portlet:createPortletFloatURL>` tag generates a webflow URL that represents creating a “floating” portlet, which is a portlet that appears in an independent window.

Table 2-10 describes the `<portlet:createPortletFloatURL>` tag attributes.

Table 2-10 `<portlet:createPortletFloatURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	The page from which the float URL is generated.	R

Table 2-10 `<portlet:createPortletFloatURL>` **Tag Attributes**

Tag Attribute	Required	Type	Description	R/C
portletName	No	String	The portlet ID. This is usually automatically generated, but can be modified.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-10 `<portlet:createPortletFloatURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-10 `<portlet:createPortletFloatURL>` **Tag Attributes**

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-10 `<portlet:createPortletFloatURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Example

Listing 2-8 illustrates how to use the `<portal:createPortletFloatURL>` JSP tag:

Listing 2-8 Using `<portlet:createPortletFloatURL>`

```
<%
        }
        if (PortletRenderHelper.isFloatable(request))
        {
            url = portletState.getUrl(Portlet.URL_CONTENT);
%>
            <util:validURL url="<%=url%>">
<% request.setAttribute( "fullscreenPortletURL", url); %>

<td width="1%" valign="middle">
   <a href="javascript:submitForm();"

onClick="openBrWindow('<portlet:createPortletFloatURL
httpsInd='http'/>','<%= portletWindowName%>',
'scrollbars=yes,width=780,height=550'); return false;">
<img src="<webflow:createResourceURL
resource='<%=imagesPath+"portlet_float.gif"%>'/>" hspace="1"
vspace="2" border="0" alt="Float" align="absmiddle"></a></td>
</util:validURL>
```

<portlet:getException>

Tag Library	`portlet.tld`
Import Statement	`<%@ taglib uri="portlet.tld" prefix= "portlet" %>`
Classes Implemented	GetExceptionTag GetExceptionTagExtraInfo

The `<portlet:getException>` tag is used to retrieve the exception or message thrown by a webflow processor. This can be the message associated with a InvalidFormFieldException or ProcessingException. This tag can be inlined in which it calls getMessage() on the exception or return a scripting variable representing the exception itself.

Table 2-11 describes the `<portlet:getException>` tag attributes.

Table 2-11 <portlet:getException> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
id	no	String	java scripting variable, can be used to retrieve an instance of the exception.	R
type	No	String	java class name, can be used to cast your exception.	R

Portal Tag Libraries

The following sections describe the portal tags and their attributes.

<portal:createWebflowURL>

Tag Library	`portal.tld`
Import Statement	`<%@ taglib uri="portal.tld" prefix= "portal" %>`
Classes Implemented	CreatePortalURLTag CreateWebflowURLTagExtraInfo

The <portal:createWebflowURL> tag is used in a JSP to dynamically create a Webflow URL in a JSP. The Webflow URL may include the protocol, domain name, port, Web application URI, WebflowServlet URI, and query string.

Table 2-12 describes the <portal:createWebflowURL> tag attributes.

Table 2-12 <portal:createWebflowURL> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	The page for which the Webflow URL is being generated.	R
origin	No	String	The node where the event will be coming from. Origins follow the node-name.node-type format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
event	yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the WebflowServlet URI. This information can then be retrieved via the request.getPathInfo() method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R

Table 2-12 `<portal:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALU`
			`E</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-12 `<portal:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-12 `<portal:createWebflowURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

<portal:form>

Tag Library	`portal.tld`
Import Statement	`<%@ taglib uri="portal.tld" prefix= "portal" %>`

Classes Implemented	PortalFormTag CreateWebflowURLTagExtraInfo

The `<portal:form>` tag is used in a JSP to dynamically generate an HTML form tag. This tag is not as sophisticated as the `<portal:validatedForm>` tag, but is simpler. For more information about the `<portal:validatedForm>` tag, refer to the next section.

Table 2-13 describes the `<portal:form>` tag attributes.

Table 2-13 `<portal:form>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	no	String	`The page for which the form is being generated.`	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
event	yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R

Table 2-13 `<portal:form>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-13 `<portal:form>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-13 `<portal:form>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-13 `<portal:form>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R
hide	No	String	If set to `false`, the namespace, origin, and event will be displayed on the command line instead of as hidden form fields. Valid values are `true` and `false`. The default value is `true`.	R
method	No	String	The method to be used for the form. Valid values are `get` and `post`. The default value is `post`.	R
name	No	String	The name of the form.	R

Table 2-13 `<portal:form>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
htmlAttributes	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R

<portal:validatedForm>

Tag Library	`portal.tld`
Import Statement	`<%@ taglib uri="portal.tld" prefix= "portal" %>`
Classes Implemented	PortalValidateFormTag WebflowValidateFormTagExtraInfo

The `<portal:validatedForm>` tag is used to dynamically generate HTML forms that can be validated. When a Web site visitor enters invalid information, the visitor's input is preserved and redisplayed with an appropriate error message.

Table 2-14 describes the `<portal:validatedForm>` Tag Attributes.

Table 2-14 `<portal:validatedForm>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	The page from which the validated form is being generated.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
event	yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R

Table 2-14 `<portal:validatedForm>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-14 `<portal:validatedForm>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VAL`
			`UE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

Table 2-14 `<portal:validatedForm>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Table 2-14 `<portal:validatedForm>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-14 `<portal:validatedForm>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
hide	No	String	If set to `false`, the namespace, origin, and event will be displayed on the command line instead of as hidden form fields. Valid values are `true` and `false`. The default value is `true`.	R
method	No		The method to be used for the form. Valid values are `get` and `post`. The default value is `post`.	R
name	No	String	The name of the form.	R
validStyle	No	String	The style used to format the HTML field when it is valid.	R
invalidStyle	No	String	The style used to format the HTML field or the message when the field is invalid.	R
unspecifiedStyle	No	String	Used to specify the initial style of the HTML field before validation occurs.	R

Table 2-14 `<portal:validatedForm>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
styleId	No	String	Scripting variable that will be set to one of invalidStyle, unSpecifiedStyle, or validStyle, depending on the field's status: `valid`, `invalid`, `unspecified`. Can be used for finer control of formatting the HTML form.	R
applyStyle	No	String	Applies the associated style as indicated by the field status to the message, the field, or to none. Therefore, valid values are `message`, `field`, and `none`. The default value is `message`.	R
messageAlign	No	String	Indicates whether to align the error message above the field, to the right of the field, or below the field. Therefore, value values are `top`, `right`, and `bottom`. The default value is `right`.	R
htmlAttributes	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R

<portal:createPortalPageChangeURL>

Tag Library	`portal.tld`

Import Statement	`<%@ taglib uri="portal.tld" prefix=`
	`"portal" %>`

Classes Implemented	CreatePageChangeURLTag

The `<portal:createPortalPageChangeURL>` tag generates a webflow URL for a page change event.

Table 2-15 describes the `<portlet:createPortalPageChangeURL>` tag attributes.

Table 2-15 `<portal:createPortalPageChangeURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
pageName	No	String	The page for which the event will generated.	R
origin	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
pathprefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathsuffix	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
extraParams	No	String	Used to supply additional request parameters as name/value pairs.	R
doRedirect	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R

Table 2-15 `<portal:createPortalPageChangeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are calculate, http, and https.
			Calculate will yield HTTPS if any node in the origin/event branch chain list is matched under the HTTPS_URL_PATTERNS context parameter in the application's WEB-INF/web.xml file. Calculate is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for HTTPSIND_DEFAULT_VALUE in web.xml is used. If no value is found there, calculate is used.
			If an invalid value is used for this attribute, calculate is used.
			The following example shows a protocol set in web.xml. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of HTTP, HTTPS or CALCULATE can be specified in all uppercase or all lowercase.
			See "Enabling HTTPS_URL_PATTERNS" on page 7-27 for more information.
				R

Table 2-15 `<portal:createPortalPageChangeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
encode	No	String	Tells Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies.See "Encoding Webflow URLs" for more information.	R

Table 2-15 `<portal:createPortalPageChangeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 2-15 `<portal:createPortalPageChangeURL>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information..	

Example

Listing 2-9 illustrates how to use the `<portal:createPortalPageChangeURL>` JSP tag:

Listing 2-9 Using <portal:createPortalPageChangeURL>

```
     if (DISPLAY_TEXT_LINKS)
        {
                    if (portalPageName.equals(selectedPage))

              {
%>
<td class="tabselected" nowrap align="center">
<b>&nbsp;
<a href="<ptl:createPortalPageChangeURL pageName='<%=
portalPageName %>'/>"><span
class="tabselected"><%=portalPageName%></span></a></b></td>
                <td class="tabselected"
WIDTH="2"><b>&nbsp;</b></td>
<%
```

```
                           }
else
                           {
%>

<td class="tabunselected" nowrap align="center">

<b>&nbsp;
<a href="<ptl:createPortalPageChangeURL pageName='<%=
portalPageName %>'/>"><span
class="tabunselected"><%=portalPageName%></span></a></b></td>
<td class="tabunselected" WIDTH="2"><b>&nbsp;</b></td>
<%            }
```

Utility Tag Libraries

The following section describes the utility tags and their attributes.

<util:validURL>

Tag Library	`util.tld`
Import Statement	`<%@ taglib uri="util.tld" prefix= "util" %>`
Classes Implemented	ValidURLTag

Processes the body if the supplied url is not null and greater than three characters long. Table 2-16 describes the `<util:validURL>` tag attributes.

Table 2-16 `<util:validURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
URL	yes	String	The supplied URL that is processed if it is valid.	R

<util:invalidURL>

Tag Library	`util.tld`
Import Statement	`<%@ taglib uri="util.tld" prefix= "util" %>`
Classes Implemented	InvalidURLTag

The `<utility:invalidURL>` tag processes the body if the supplied url is null or less than four characters long. Table 2-17 describes the `<util:invalidURL>` tag attributes.

Table 2-17 `<util:invalidURL>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
URL	yes	String	The supplied URL.	R

Example

The following code sample illustrates how to use the `<utility:invalidURL>` as well as the <`utility:validURL`> JSP tag:

Listing 2-10 Using `<utility:invalidURL>` and `<utility:validURL>`

```
<%

// First try for an alternate header if one doesn't exist use
// the regular header

url =  portletState.getUrl(Portlet.URL_ALTERNATE_HEADER);
   debug.out("Alternate header: " + url);
%>
<util:invalidURL url="<%=url %>">
   <% url = portletState.getUrl(Portlet.URL_HEADER); %>
</util:invalidURL>
<util:validURL url="<%=url %>">
   <table cellpadding="4" cellspacing="0" width="100%"
   class="portletheader">
      <tr>
         <td width="100%">
            <jsp:include page="<%=url%>"/>
         </td>
      </tr>
   </table>
</util:validURL>

<%
```

3 Personalization JSP Tags

The JSP tags included with WebLogic Portal allow developers to create personalized applications without having to program using Java.

This section includes information on the following JSP tags:

- Ads
- Content Management
- Internationalization
- Personalization Tags
- Placeholders
- Property Sets
- User Management: Profile Management Tags
- User Management: Group-User Management Tags
- User Management: Security Tags
- Utility Tags: Personalization Utilities
- Utility Tags: WebLogic Utilities

Ads

The Ad tag queries the content management system and displays ads.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<ad:adTarget>

Tag Library	`ad.tld`
Import Statement	`<%@ taglib uri="ad.tld" prefix="ad" %>`
Classes Implemented	adTargetTag

The `<ad:adTarget>` (Table 3-1) uses the Ad Service to send an ad query to the content management system. Unlike the <ph:placeholder> tag, the query in the `<ad:adTarget>` tag does not compete with other queries in an ad placeholder.

Use this tag if you need to make sure that a given ad displays to customers in a specific location. Depending on how narrowly you construct the query, you might have to remove or modify this tag when you want to display a different ad.

If the ad query returns more than one ad, the Ad Service uses the `adWeight` attribute of each ad to determine which ad to display.

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	SPI
1. `AdService.getContent()`	3. `DocumentProvider.findDocumentMetadata()`
2. `DocumentManager.getContent()`	4. `DocumentMetadataDef.getID()`
8. `Document.getProperty()` ... (other `get*` methods)	5. `DocumentMetadataDef.getName()` ... (other `get*` methods)
9. `Document.getContent()`	6. `DocumentMetadataDef.getAttributeNames()`
10. `DocumentManager.getContentBlock()`	7. `DocumentMetadataDef.getAttribute()`
	11. `DocumentProvider.getDocument()`
	12. `DocumentDef.openStream()`

Tag Attributes

Table 3-1 describes the <ad:adTarget> tag attributes.

Table 3-1 <ad:adTarget> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
query	Yes	String	Contains a query that the Ad Service uses to find content. Use the query syntax described in the *Javadoc* for `com.beasys.commerce.util.ExpressionHelper`. For details on constructing queries, see "Constructing Content Queries" in the *Development Guide* at http://edocs.bea.com/wlp/docs70/dev/conmgmt.htm.	R

Table 3-1 <ad:adTarget> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
height	No	int	Specifies the height (in pixels) that the placeholder uses when generating the HTML that the browser requires to display a document. The placeholder uses this value only for content types to which display dimensions apply and only if other attributes have not already defined dimensions for a given document. If you do not specify this value and other attributes have not already been defined, the browser behavior determines the height of the document.	R
width	No	int	Specifies the width (in pixels) that the placeholder uses when generating the HTML that the browser requires to display a document. The placeholder uses this value only for content types to which display dimensions apply and only if other attributes have not already defined dimensions for a given document. If you do not specify this value and other attributes have not already been defined, the browser behavior determines the width of the document.	R

Example

Listing 3-1 picks one of the ads in the ad group "Car" and renders it in a space measuring 200 x 400 pixels.

Listing 3-1 Using `<ad:adTarget>`

```
<%@ taglib uri="ad.tld" prefix="ad" %>
.
```

```
.
.
<ad:adTarget query="group == 'ads'" />
```

Content Management

The Content Management component includes four JSP tags. These tags allow a JSP developer to include non-personalized content in a HTML-based page. The `cm:select` and `cm:selectbyid` tags support content caching for content searches. Note that none of the tags support or use a body.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<cm:getProperty>

Tag Library	`cm.tld`
Import Statement	`<%@ taglib uri="cm.tld" prefix="cm" %>`
Classes Implemented	getPropertyTag getPropertyExtraInfo

The `<cm:getProperty>` tag retrieves the value of the specified content metadata property into a variable specified by `resultId`. It does not have a body. If `resultId` is not specified, the value will be inlined into the page, similar to the `<cm:printProperty>` tag. This tag operates on any ConfigurableEntity, not just the Content object. However, it does not support ConfigurableEntity successors.

How the Tag Maps to the Content Management Service Provider Interface

Because this tag receives data that has already been retrieved from the content management system, it makes no calls to the content management service provider interface (SPI).

Tag Attributes

Table 3-2 describes the `<cm:getProperty>` tag attributes.

Table 3-2 <cm:getProperty>

Tag Attribute	Required	Type	Description	R/C
id	No	String	The JSP script variable name which contains the Content instance from which to get the properties.	R
entity	No	ConfigurableEntity	Specifies the com.beasys.commerce.foundation. ConfigurableEntity object from which to get the property. If this is specified and non-null, `id` is ignored. Otherwise, `id` will be used.	R
name	Yes	String	The name of the property to print.	R
scope	No	String	The scope name for the property to get. If not specified, null is passed in, which is what Document objects expect.	R
resultId	no	String	The name of the JSP script variable which will be populated with the value of the property. If this is not specified, then the value of the property will be inlined into the body of the JSP. If this is specified, then `encode`, `default`, `maxLength`, `dateFormat`, and `numFormat` are ignored.	C
resultType	no	String	The Java type of the property. If this is not specified, then `java.lang.Object` is used.	C

Table 3-2 <cm:getProperty> (Continued)

Tag Attribute	Required	Type	Description	R/C
encode	No	String	Either *html*, *url*, or *none*: ■ If *html*, then the value will be html encoded so that it appears in HTML as expected (& becomes &amp;, < becomes &lt;, > becomes &gt;, and " becomes &quot;). ■ If *url*, then it is encoded to x-www-form-urlencoded format via the java.net.URLEncoder. ■ If *none* or unspecified, no encoding is performed.	R
default	No	String	The value to print if the property is not found or has a null value. If this is not specified and the property value is null, nothing is printed.	R
maxLength	No	String, int	The maximum length of the property's value to print. If specified, values longer than this will be truncated.	R
failOnError	No	String, Boolean	This attribute can have one of two values: `False` (default value): Handles JSP processing errors gracefully and prints nothing if an error occurs. `True`: Throws an exception. You can handle the exception in the code, let the page proceed to the normal error page, or let the application server handle it less gracefully.	R
dateFormat	No	String	The java.text.SimpleDateFormat string to use to print the property, if it is a java.util.Date. If the property is not a Date, this is ignored. If this is not set, the Date's default `toString` method is used.	R

Table 3-2 <cm:getProperty> (Continued)

Tag Attribute	Required	Type	Description	R/C
numFormat	No	String	The java.text.DecimalFormat string to use to print the property, if it is a java.lang.Number. If the property is not a Number, this is ignored. If this is not set, the Number's default `toString` method is used.	R

Example

Listing 3-2 shows how to use `<cm:getProperty>` to get the String value of the `name` property from the Content object stored at `doc` and place it in the `contentName` variable.

Listing 3-2 Using `<cm:getProperty>`

```
<%@ taglib uri="cm.tld" prefix="cm" %>
.
.
.
<cm:getProperty resultId="contentName" resultType="String"
 id="content" name="name" />
<es:notNull item="<%=contentName%>">
The name is not null.
</es:notNull>
```

<cm:printDoc>

Tag Library	`cm.tld`
Import Statement	`<%@ taglib uri="cm.tld" prefix="cm" %>`
Classes Implemented	printDocTag

The `<cm:printDoc>` tag inlines the raw bytes of a `Document` object into the JSP output stream. This tag does not support a body and only supports `Document` objects. It does not differentiate between text and binary data.

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	SPI
1. `Document.getIdentifier()` 2. `Document.getPropertyAsString()` 3. `Document.getContent()` 4. `DocumentManager.getContentBlock()` 7. `Document.getSize()`	5. `DocumentProvider.getDocument()` 6. `DocumentDef.openStream()`

Tag Attributes

Table 3-3 describes the `<cm:printDoc>` tag attributes.

Table 3-3 <cm:printDoc> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
id	No	String	The JSP script variable name which contains the Content instance from which to get the properties.	R
blockSize	No	String, int	The size of the blocks of data to read. The default is 8K. Use 0 or less to read the entire block of bytes in one operation.	R
start	No	String, int	Specifies the index in the bytes where to start reading. Defaults to 0.	R
end	No	String, int	Specifies the index in the bytes where to stop reading. The default is to read to the end of the bytes.	R

Table 3-3 <cm:printDoc> Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
encode	No	String	Either html, url, or none: ■ If html, then the value will be html encoded so that it appears in HTML as expected (& becomes &amp;, < becomes &lt;, > becomes &gt;, and " becomes &quot;). ■ If url, then it is encoded to x-www-form-urlencoded format via the java.net.URLEncoder. ■ If none or unspecified, no encoding is performed.	R
document	No	Document	Specifies the com.bea.p13n.content.document.Document to use. If this is specified and non-null, `id` will be ignored. Otherwise, `id` will be used.	R
failOnError	No	String, Boolean	This attribute can have one of two values: `False` (default value): Handles JSP processing errors gracefully and prints nothing if an error occurs. `True`: Throws an exception. You can handle the exception in the code, let the page proceed to the normal error page, or let the application server handle it less gracefully.	R
baseHref	No	String	The URL of the document's BASE HREF. This can be either an absolute URL or a relative URL.	R

Note: If `baseHref` is provided, then the `<cm:printDoc>` tag will output a starting <BASE HREF> using the value of the `baseHref` parameter. If `baseHref` is not a fully complete URL, the missing parts will be filled in based upon the URL of the outermost page.

Additionally, if `baseHref` is provided, then, after printing the document, the `<cm:printDoc>` tag will output a <BASE HREF> based upon the URL of the outermost page.

Example

Listing 3-3 shows how to use `<cm:printDoc>` to get a Document object from an `id` in the `request` attributes and inline the Document's text (which might contain relative links).

Listing 3-3 Using `<cm:printDoc>`

```
<%@ taglib uri="cm.tld" prefix="cm" %>
.
.
.
.<% String contentId = request.getParameter("contentId"); %>
<cm:selectById contentId="<%=contentId%>" id="doc" />
<cm:printDoc id="doc" blockSize="1000" baseHref="/ShowDocServlet"
/>
```

<cm:printProperty>

Tag Library	`cm.tld`
Import Statement	`<%@ taglib uri="cm.tld" prefix="cm" %>`
Classes Implemented	printPropertyTag

The `<cm:printProperty>` tag inlines the value of the specified content metadata property as a string. It does not have a body. This tag operates on any `ConfigurableEntity`, not just the `Content` object. However, it does not support `ConfigurableEntity` successors.

How the Tag Maps to the Content Management Service Provider Interface

Because this tag receives data that has already been retrieved from the content management system, it makes no calls to the content management service provider interface (SPI).

Tag Attributes

Table 3-4 describes the `<cm:printProperty>` tag attributes.

Table 3-4 <cm:printProperty>

Tag Attribute	Required	Type	Description	R/C
id	No	String	The JSP script variable name which contains the Content instance from which to get the properties.	R
name	Yes	String	The name of the property to print.	R
entity	No	ConfigurableEntity	Specifies the com.beasys.commerce.foundation. ConfigurableEntity object from which to get the property. If this is specified and non-null, `id` is ignored. Otherwise, `id` will be used.	R
scope	No	String	The scope name for the property to get. If not specified, null is passed in, which is what Document objects expect.	R
encode	No	String	Either *html*, *url*, or *none*: ■ If *html*, then the value will be html encoded so that it appears in HTML as expected (& becomes &amp;, < becomes &lt;, > becomes &gt;, and ” becomes &quot;). ■ If *url*, then it is encoded to x-www-form-url encoded format via the java.net.URLEncoder. ■ If *none* or unspecified, no encoding is performed.	R
default	No	String	The value to print if the property is not found or has a null value. If this is not specified and the property value is null, nothing is printed.	R
maxLength	No	String, int	The maximum length of the property’s value to print. If specified, values longer than this will be truncated.	R

Table 3-4 <cm:printProperty> (Continued)

Tag Attribute	Required	Type	Description	R/C
failOnError	No	String, Boolean	This attribute can have one of two values: `False` (default value): Handles JSP processing errors gracefully and prints nothing if an error occurs. `True`: Throws an exception. You can handle the exception in the code, let the page proceed to the normal error page, or let the application server handle it less gracefully.	R
dateFormat	No	String	The java.text.SimpleDateFormat string to use to print the property, if it is a java.util.Date. If the property is not a Date, this is ignored. If this is not set, the Date's default `toString` method is used.	R
numFormat	No	String	The java.text.DecimalFormat string to use to print the property, if it is a java.lang.Number. If the property is not a Number, this is ignored. If this is not set, the Number's default `toString` method is used.	R

Example

Listing 3-4 shows how to use `<cm:printProperty>` to have a text input field’s default value be the first 75 characters of the subject of a `Content` object stored at *`doc`*:

Listing 3-4 Using `<cm:printProperty>`

```
<%@ taglib uri="cm.tld" prefix="cm" %>
.
.
.
<form action=”javascript:void(0)”>
   Subject: <input type=”text” size=”75” name=”subject”
   value=”<cm:printProperty id=”doc” name=”Subject” maxLength=”75”
   encode=”html”/>” >
</form>
```

<cm:select>

Tag Library	`cm.tld`
Import Statement	`<%@ taglib uri="cm.tld" prefix="cm" %>`
Classes Implemented	selectTag selectExtraInfo

This tag uses only the search expression query syntax to select content. It does not support or use a body. After this tag has returned the `<es:forEachInArray>` tag (see "<es:forEachInArray>" on page 3-96,) zero can be used to iterate over the array of `Content` objects. This tag supports generic `Content` via a `ContentManager` interface.

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	**SPI**
1. `DocumentManager.getContent()`	2. `DocumentProvider.findDocumentMetadata()` 3. `DocumentMetadataDef.getID()` 4. `DocumentMetadataDef.getName()` ... (other `get*` methods) 5. `DocumentMetadataDef.getAttributeNames()` 6. `DocumentMetadataDef.getAttribute()`

Tag Attributes

Table 3-5 describes the `<cm:select>` tag attributes.

Table 3-5 <cm:select> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
contentHome	No	String	The JNDI name of the ContentManager EJB Home to use to find content. The object in JNDI at this name must implement a `create` method which returns an object which implements the ContentManager interface. If not specified, the system searches the default content home.	R
max	No	String, long	Limits the maximum number of content items returned. If not present, or zero or less, it returns all of the content items found.	R

sortBy	No	String	A list of document attributes by which to sort the content. The syntax follows the SQL *order by* clause. The sort specification is limited to a list of the metadata attribute names and the keywords `ASC` and `DESC`.
			Examples:
			sortBy="creationDate"
			sortBy="creationDate ASC, title DESC"
				R

failOnError	No	String, Boolean	This attribute can have one of two values: `False` (default value): Handles JSP processing errors gracefully and returns an empty array if an error occurs. `True`: Throws an exception that causes the JSP page to stop. You can handle the exception in the code, let the page proceed to the normal error page, or let the application server handle it less gracefully.	R
id	Yes	String	The JSP script variable name that will contain the array of Content objects after this tag finishes.	C

Table 3-5 <cm:select> Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
query	No	String	A content query string used to search for content. Example: query="mimetype contains 'text' && author='Proulx'". For details on constructing queries, see “Constructing Content Queries” in the *Development Guide* at http://edocs.bea.com/wlp/docs70/dev/conmgmt.htm.	R
expression	No	Expression	The com.beasys.commerce.foundation.expression.Expression object to use to search for content. If this is null or not specified, then `query` must be specified. Otherwise, `query` is ignored.	R
useCache	No	String, Boolean	Determines whether Content is cached. This attribute can have one of two values: `False` (default value): ContentCache is not used. If `false` (not specified), the `cacheId`, `cacheScope` and `cacheTimeout` settings are ignored. `True`: ContentCache is used.	R
cacheId	No	String	The identifier name used to cache the Content. Internally, the cache is implemented as a Map; this will become the key. If not specified, the `id` attribute of the tag is used.	R
cacheTimeout	No	String, long	The time, in milliseconds, for which the cached Content is valid. If more than this amount of time has passed since the Content was cached, the cached Content will be cleared, retrieved, and placed back into the cache. Use -1 for no-timeout (always use the cached Content). Default = -1.	R

Table 3-5 <cm:select> Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
cacheScope	No	String	Specifies the lifecycle scope of the content cache. Similar to `<jsp:useBean>`. Possible values: ■ `application` ■ `session` (the default) ■ `page` ■ `request`	R
contextParams	No	String or java.util.Map	Additional search parameters to pass to the ContentManager. Some ContentManager implementations may support this.	R
readOnly	Ignored		This attribute is deprecated and no longer used. When found, it is ignored.	

Example

Listing 3-5 shows how to use `<cm:select>` to find the first five text `Content` objects that are marked as news items for the evening using the ContentCache, and print out the titles in a list:

Listing 3-5 Using <cm:select>

```
<%@ taglib uri="cm.tld" prefix="cm" %>
.
.
.
<cm:select
contentHome="<%=ContentHelper.DEF_CONTENT_MANAGER_HOME%>" max="5"
useCache="true" cacheTimeout="300000" cacheId="Evening News"
sortBy="creationDate ASC, title ASC" query="
    type = ‘News’ && timeOfDay = ‘Evening’ && mimetype like
    ‘text/*’ " id="newsList"/>
```

```
<ul>
   <es:forEachInArray array="<%=newsList%>" id="newsItem"
   type="com.bea.p13n.content.Content">
       <li><cm:printProperty id="newsItem" name="Title"
       encode="html" />
   </es:forEachInArray>
</ul>
```

<cm:selectById>

Tag Library	`cm.tld`
Import Statement	`<%@ taglib uri="cm.tld" prefix="cm" %>`
Classes Implemented	selectByIDTag selectByIDExtraInfo

The `<cm:selectById>` tag (Table 3-6) retrieves content using the `Content's` unique identifier. This tag does not have a body. This tag is basically a wrapper around the `select` tag. It works against any `Content` object which has a string-capable primary key.

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	SPI
1. `DocumentManager.getContent()`	2. `DocumentProvider.findDocumentMetadata()` 3. `DocumentMetadataDef.getID()` 4. `DocumentMetadataDef.getName()` ... (other `get*` methods) 5. `DocumentMetadataDef.getAttributeNames()` 6. `DocumentMetadataDef.getAttribute()`

Tag Attributes

Table 3-6 describes the `<cm:selectById>` tag attributes.

Table 3-6 <cm:selectById>

Tag Attribute	Required	Type	Description	R/C
contentHome	No	String	The JNDI name of the ContentManager EJB Home to use to find content. The object in JNDI at this name must implement a `create` method which returns an object that implements the ContentManager interface. If not specified, the system searches the default content home.	R
contentId	Yes	String	The string identifier of the piece of content.	R
onNotFound	No	String	If the content object specified by `contentId` cannot be found, this controls the behavior. If this is set, then an Exception will be thrown with the value as the message; if this is not set, the tag will return `null`.	R

Table 3-6 <cm:selectById> (Continued)

Tag Attribute	Required	Type	Description	R/C
failOnError	No	String, Boolean	This attribute can have one of two values: `False` (default value): Handles JSP processing errors gracefully and returns null if an error occurs. `True`: Throws an exception that causes the JSP page to stop. You can handle the exception in the code, let the page proceed to the normal error page, or let the application server handle it less gracefully.	R
id	Yes	String	The JSP script variable name that contains the Content object after this tag finishes. If the Content object with the specified identifier does not exist, it contains null.	C
useCache	No	String, Boolean	Determines whether Content is cached. This attribute can have one of two values: `False` (default value): ContentCache is not used. If `false` (not specified), the `cacheId`, `cacheScope` and `cacheTimeout` settings are ignored. `True`: ContentCache is used.	R
cacheId	No	String	The identifier name used to cache the Content. Internally, the cache is implemented as a Map; this will become the key. If not specified, the `id` attribute of the tag is used.	R
cacheTimeout	No	String, long	The time, in milliseconds, for which the cached Content is valid. If more than this amount of time has passed since the Content was cached, the cached Content will be cleared, retrieved, and placed back into the cache. Use -1 for no-timeout (always use the cached Content). Default = -1.	R

Table 3-6 <cm:selectById> (Continued)

Tag Attribute	Required	Type	Description	R/C
cacheScope	No	String	Specifies the lifecycle scope of the content cache. Similar to `<jsp:useBean>`. Possible values: ■ `application` ■ `session` (the default) ■ `page` ■ `request`	R
contextParams	No	String or java.util.Map	Additional search parameters to pass to the ContentManager. Some ContentManager implementations may support this.	R
readOnly	Ignored		This attribute is deprecated and no longer used. When found, it is ignored.	

Example

Listing 3-6 shows how to use `<cm:selectById >` to fetch the `Document` (using ContentCaching) with an identifier of *`1234`* and inline its content.

Listing 3-6 Using `<cm:selectById >`

```
<%@ taglib uri="cm.tld" prefix="cm" %>
.
.
.
<cm:selectById
contentHome="<%=ContentHelper.DEF_CONTENT_MANAGER_HOME%>"
contentId="contentportlet/sports1.htm"
id="doc" useCache="true" cacheTimeout="300000" cacheId="1234" />
<cm:printDoc id="doc" />
```

Internationalization

These tags are used in the localization of JSP pages that have an internationalization requirement.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

Note: With the `<i18n>` tags, you can point to resource bundle files that contain localized content for display in your JSPs. For information on setting the interval at which WebLogic Portal checks for updated resource bundle content, see "Internationalization Performance Tuning" in the *Administration Guide* at http://edocs.bea.com/wlp/docs70/admin/sysadmin.htm.

<i18n:localize>

Tag Library	`i18n.tld`
Import Statement	`<%@ taglib uri="i18n.tld" prefix="i18n" %>`
Classes Implemented	LocalizeTag LocalizeExtraInfo

This tag allows you to define the language, country, variant, and base bundle name to be used throughout a page when accessing resource bundles via the `<i18n:getmessage>` tag.

This tag also specifies a character encoding and content type to be specified for a JSP page. Because of this, the tag should be used as early in the page as possible—before anything is written to the output stream—so that the bytes are properly encoded.

When an HTML page is included in a larger page, only the larger page can use the `<i18n:localize>` tag. This is because the `<i18n:localize>` tag sets the encoding for the page, and the encoding must be set in the parent (including) page before any bytes are written to the response's output stream. The parent page must set an encoding that is sufficient for all the content on that page as well as any included pages.

The preferred approach is to retrieve all strings dynamically from the `<i18n:getMessage>` tag, and avoid embedding strings statically (that is, avoid hard-coding them) in your JSP page.

If your page contains only dynamic strings (strings retrieved using the `<i18n:getMessage tag>`), then do not use the `<i18n:localize>` tag in conjunction with the `<%@ page contentType="<something>" >` page directive defined in the JSP specification. The directive is unnecessary if you are using the `<i18n:localize>` tag, and can result in inconsistent or wrong `contentType` declarations.

If you must mix static strings and dynamic strings on the same page, then you will need to use the page directive. Ensure that the character set specified by the `<i18n:localize>` tag is compatible with the character set specified in the page directive.

Tag Attributes

Table 3-7 describes the `<i18n:localize>` tag attributes.

Table 3-7 `<i18n:localize>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
bundleName	No	String	The base name of the MessageBundle is used to retrieve localized text for a JSP page. For information on setting the interval at which WebLogic Portal checks for updated resource bundle content, see "Internationalization Performance Tuning" in the *Administration Guide* at http://edocs.bea.com/wlp/docs70/admin/sysadmin.htm.	R

Table 3-7 `<i18n:localize>` (Continued)Tag Attributes

Tag Attribute	Required	Type	Description	R/C
language	No	String or String `[]`	A String—two character ISO Language Code—denoting the user's preferred language, or a String `[]` containing a list of preferred language codes for a user, with stronger preferences indexed lower (earlier) in the array.	R
country	No	String	The two character ISO Country Code for a country. For example, this code would be used to look for a MessageBundle containing text localized to English speaking users in the U.S. as opposed to English speaking users in the U.K.	R
variant	No	String	A String representing a locale's variant. The variant is used when localization demands a more specific locale than can be denoted by having just language and a country.	R
locale	No	java.util.Locale	Instead of specifying language, country, and variant as Strings, a `java.util.Locale` object can be provided. If provided, the values in the Locale's language, country, and variant fields will negate any of the other language, country, and variant values passed to the tag as Strings.	R
charset	No	String	The name of the character encoding set to use for this page. Defaults to "UTF-8" if no encoding can be determined for the chosen language, otherwise an encoding appropriate for the chosen language is used.	R
contentType	No	String	The type of content contained in the page, defaults to "text/html".	R

Example 1

Listing 3-7 shows how to use `<i18n:localize>` to define a single language preference.

Listing 3-7 Using `<i18n:localize>`; Example 1

```
<%@ taglib uri="i18n.tld" prefix="i18n" %>
.
.
.
<%
// Definition of a single language preference
String language = "en";
%>

<i18n:localize language="<%=language%>"
bundleName="i18nExampleResourceBundle"/>
<html>
<body>
<i18n:getMessage messageName="greeting"/>
</body>
</html>
```

Example 2

Listing 3-8 shows how to use `<i18n:localize>` to define two language preferences, English and Spanish.

Listing 3-8 Using `<i18n:localize>`; Example 2

```
<%@ taglib uri="i18n.tld" prefix="i18n" %>
.
.
.
<%
// Array that defines two languages preferences - English and
// Spanish in that order of preference.
```

```
String[] languages = new String[] { "en", "es" };

%>

<i18n:localize language="<%=languages%>"
bundleName="i18nExampleResourceBundle"/>
<html>
<body>
<i18n:getMessage messageName="greeting"/>
</body>
</html>
```

<i18n:getMessage>

Tag Library	`i18n.tld`
Import Statement	`<%@ taglib uri="i18n.tld" prefix="i18n" %>`
Classes Implemented	GetMessageTag GetMessageExtraInfo

This tag is used in conjunction with the `<i18n:localize>` tag to retrieve localized static text or messages from a JspMessageBundle.

Tag Attributes

Table 3-8 describes the `<i18n:getMessage>` tag attributes.

Table 3-8 <i18n:getMessage> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
id	No	String	Holds the value of the label (or message) in the JSP page.	C
messageName	Yes	String	The key for the message bundle.	R

Table 3-8 <i18n:getMessage> Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
messageArgs	No	Object []	The arguments to the message bundle. If no args are provided, it is assumed that static text (not a message) is to be returned. For example, {"Wednesday", "78"}; might be used to construct the message “Today is Wednesday, and the temperature is 78 degrees Fahrenheit.”	R
bundleName	No	String	If properly initialized in the `<i18n:localize>` tag, there is no need to pass this tag attribute unless it is desired to use a different bundle for a particular tag invocation. For information on setting the interval at which WebLogic Portal checks for updated resource bundle content, see “Internationalization Performance Tuning” in the *Administration Guide* at http://edocs.bea.com/wlp/docs70/admin/sysadmin.htm.	R
language	No	String	If properly initialized in the `<i18n:localize>` tag, there is no need to pass this tag attribute, unless it is desired to use a different language for a particular tag invocation.	R
country	No	String	If properly initialized in the `<i18n:localize>` tag, there is no need to pass this tag attribute, unless it is desired to use a different country for a particular tag invocation.	R
variant	No	String	If properly initialized in the `<i18n:localize>` tag, there is no need to pass this tag attribute, unless it is desired to use a different variant for a particular tag invocation.	R

Table 3-8 <i18n:getMessage> Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
locale	No	java.util.Locale	If properly initialized in the `<i18n:localize>` tag, there is no need to pass this tag attribute, unless it is desired to use a different locale (language, country, and variant) for a particular tag invocation.	R

Example 1

Listing 3-9 produces this output:

```
Welcome To This Page! 14 out of 100 files have been saved
```

when included in a JSP.

Listing 3-9

```
<%@ taglib uri="i18n.tld" prefix="i18n" %>
.
.
.
<%
// Definition of a single language preference
String language = "en";

// Creation of message arguments
Object[] args = new Object[]
{
    new Integer(14),
    new Integer(100)
};
%>

<i18n:localize language="<%=language%>"
bundleName="i18nExampleResourceBundle"/>
<html>
<body>
<i18n:getMessage messageName="greeting"/>
<i18n:getMessage messageName="message" messageArgs="<%=args%>"/>
```

```
</body>
</html>
```

The following code shows the entries in the property file named `i18nExampleResourceBundle. properties`:

```
greeting=Welcome To This Page!
  message={0} out of {1} files have been saved.
```

Personalization Tags

The <pz:div> tag, <pz:contentSelector> tag, and <pz:contentQuery> tag use the Advisor to classify the user, select content, and retrieve content, respectively.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

This section contains information on the following subjects:

- pz Tags and the Internal Cache
- <pz:contentQuery>
- <pz:contentSelector>
- <pz:div>

pz Tags and the Internal Cache

Content search `contextParams` support per-search configuration attributes and can be used to determine whether to use the internal cache. The <`cm:select`> and <`cm:selectById`> tags support setting the `contextParams`, but the `pz` tags do not. In order to control whether a <`pz:contentSelector`> uses the internal cache, use the following approach.

Add the following to a <pz:content*> tag:

```
contextParams="someName=someValue"
```

A runtime expression like the following should be used:

```
contextParams='<%="aName=aValue bName=bValue cName=cValue"%>'
```

<pz:contentQuery>

Tag Library	pz.tld
Import Statement	<%@ taglib uri="pz.tld" prefix="pz" %>
Classes Implemented	ContentQueryTag ContentQueryExtraInfo

The <pz:contentQuery> tag performs a content attribute search for content in a content manager. If the useCache attribute is set to true, the results of a content management query will be cached. The tag only has a begin tag and does not have a body or end tag. It returns an array of Content objects returned from the content manager as the result of executing the content query.

Personalization content tags required for JSP developers to access the Content object returned might include:

An object array iterator tag. This tag provides a way to iterate over the Content objects in the array. Use the <es:forEachInArray> tag to iterate over an array of Objects. (See "<es:forEachInArray>" on page 3-96 for more information.)

- Content access tags. Content tags access metadata attributes in the content, retrieve content, and put it into the HTTP response output stream. (See the section "Content Management" on page 3-5 for more information.)

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	SPI
1. `EJBAdvisor.getAdvice()` 2. `ContentQueryAdvislet.getAdvice()` 3. `DocumentManager.getContent()`	4. `DocumentProvider.findDocumentMetadata()` 5. `DocumentMetadataDef.getID()` 6. `DocumentMetadataDef.getName()` ... (other `get*` methods) 7. `DocumentMetadataDef.getAttributeNames()` 8. `DocumentMetadataDef.getAttribute()`

Tag Attributes

Table 3-9 describes the `<pz:contentQuery>` tag attributes.

Table 3-9 `<pz:contentQuery>` Tag Attributes

Tag Attribute	Required	Type	Description	R/C
max	No	String, long	Limits the maximum number of content items returned. If not present, it returns all of the content items found.	R

sortBy	No	String	A list of document attributes by which to sort the content. The syntax follows the SQL *order by* clause. The sort specification is limited to a list of the metadata attribute names and the keywords `ASC` and `DESC`.
			Examples:
			sortBy="creationDate"
			sortBy="creationDate ASC, title DESC"
				R

Table 3-9 `<pz:contentQuery>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
query	Yes	String	A content query string used to search for content. Example: query= “mimetype contains ‘text’ && author=‘Proulx’”. For details on constructing queries, see “Constructing Content Queries” in the *Development Guide* at http://edocs.bea.com/wlp/docs70/dev/conmgmt.htm.	R
contentHome	Yes	String	The JNDI name of the ContentManager EJB Home. The object in the JNDI at this name must implement a `create` method which returns an object which implements the ContentManager interface. For more information, see the section “Specify a Value for contentHome” on page 3-38.	R
id	Yes	String	The array variable name that contains the content objects found. If no content is found, the variable is assigned an empty array (not null) of Content objects.	C
useCache	No	String, Boolean	Determines whether Content is cached. This attribute can have one of two values: `False` (default value): ContentCache is not used. If `false` (not specified), the `cacheId`, `cacheScope` and `cacheTimeout` settings are ignored. `True`: ContentCache is used.	R
cacheId	No	String	The identifier name used to cache the Content. Internally, the cache is implemented as a Map; this will become the key. If not specified, the `id` attribute of the tag is used.	R

Table 3-9 `<pz:contentQuery>` Tag Attributes (Continued)

Tag Attribute	Required	Type	Description	R/C
cacheTimeout	No	String, long	The time, in milliseconds, for which the cached Content is valid. If more than this amount of time has passed since the Content was cached, the cached Content will be cleared, retrieved, and placed back into the cache. Use -1 for no-timeout (always use the cached Content). Default = -1.	R
cacheScope	No	String	Specifies the lifecycle scope of the content cache. Similar to `<jsp:useBean>`. Possible values: ■ `application`. Any JSP page in the web application that any customer requests can access the cache. ■ `session` (the default). Any JSP in the web application that the current customer requests can access the cache. ■ `page`. Only the current JSP that any customer requests can access the cache. ■ `request`. Only the current user request can access the cache. If a customer re-requests the page, the content selector re-runs the query and recreates the cache.	R

Example

Listing 3-10 shows how to use `<pz:contentQuery>`.

Listing 3-10 Using `<pz:contentQuery>`

```
<%@ page import="bea.p13n.content.ContentHelper" %>
<%@ taglib uri="es.tld" prefix="es" %>
<%@ taglib uri="cm.tld" prefix="cm" %>
<%@ taglib uri="pz.tld" prefix="pz" %>
.
```

```
.
.
<pz:contentQuery id="docs"
contentHome="<%=ContentHelper.DEF_DOCUMENT_MANAGER_HOME%>"
   query="author = 'Hemingway'" />

<ul>


 <es:forEachInArray array="<%=docs%>" id="aDoc"
   type="com.bea.p13n.content.Content">
      <li>The document title is: <cm:printProperty id="aDoc"
      name="Title" encode="html" />
  </es:forEachInArray>
</ul>
```

<pz:contentSelector>

Tag Library	`pz.tld`
Import Statement	`<%@ taglib uri="pz.tld" prefix="pz" %>`
Classes Implemented	ContentSelectorTag ContentSelectorExtraInfo

The `<pz:contentSelector>` tag allows arbitrary personalized content to be recommended based on a content selector rule.

A content selector rule first evaluates a set of conditions that you define in the E-Business Control Center (for example, whether or not a user fits a specified classification). If the conditions evaluate to true, content is retrieved from the Content Manager based on a content query defined in the E-Business Control Center.

Note: Rules are created in the E-Business Control Center. This GUI tool is designed to allow Business Analysts to develop their own customer segments. Because the Business Analysts are not exposed to the concept of rules, you will see content selector rules called simply "content selectors" and classifier rules referred to as "customer segmentation."

To cache the results of the content selector rule, set the `useCache` attribute to `true`. If the cache has not timed out, subsequent calls to the `<pz:contentSelector>` tag will return the cached results without re-evaluating the content query.

The `<pz:contentSelector>` tag only has a begin tag and does not have a body or end tag. It returns an array of `Content` objects returned from the Content Manager as a result of executing the content query.

Tags possibly required for JSP developers to access the `Content` objects returned might include:

- An object array iterator tag. This tag provides a way to iterate over the `Content` objects in the array. Use the `<es:forEachInArray>` tag to iterate over an array of `Objects`.
- Content access tags. Content tags access metadata attributes in the content and retrieve content and put it into the HTTP response output stream.

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	SPI
1. `EJBAdvisor.getAdvice()` 2. `ContentQueryAdvislet.getAdvice()` 3. `DocumentManager.getContent()`	4. `DocumentProvider.findDocumentMetadata()` 5. `DocumentMetadataDef.getID()` 6. `DocumentMetadataDef.getName()` ... (other `get*` methods) 7. `DocumentMetadataDef.getAttributeNames()` 8. `DocumentMetadataDef.getAttribute()`

Tag Attributes

Table 3-10 describes the `<pz:contentSelector>` tag attributes.

Table 3-10 <pz:contentSelector> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
rule	Yes	String	The name of the content selector in the content selector definitions created in the E-Business Control Center.	R
contentHome	Yes	String	The JNDI name of the ContentManager EJB Home. The object in the JNDI at this name must implement a `create` method which returns an object which implements the ContentManager interface. For more information, see the section "Specify a Value for contentHome" on page 3-38.	R
max	No	String, long	Limits the maximum number of content items returned. If not present, or if equal to -1L, it returns all of the content items found.	R

sortBy	No	String	A list of document attributes by which to sort the content. The syntax follows the SQL *order by* clause. The sort specification is limited to a list of the metadata attribute names and the keywords `ASC` and `DESC`.
			Examples:
			sortBy="creationDate"
			sortBy="creationDate ASC, title DESC"
				R

Table 3-10 <pz:contentSelector> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
query	No	String	A content query string used to search for content. This query overrides any query that a Business Analyst creates in the E-Business Control Center. Example: query="mimetype contains 'text' && author='Salinger'". For details on constructing queries, see "Constructing Content Queries" in the *Development Guide* at http://edocs.bea.com/wlp/docs70/dev/conmgmt.htm.	R
id	Yes	String	The array variable name that contains the content objects found. If no content is found, the variable is assigned an empty array (not null) of Content objects.	C
useCache	No	String, Boolean	Determines whether Content is cached. This attribute can have one of two values: `False` (default value): The Content cache is not used. If `false` (not specified), the cacheId, cacheScope and cacheTimeout settings are ignored. `True`: Content cache is used.	R
cacheId	No	String	The identifier name used to cache the Content. Internally, the cache is implemented as a Map; this will become the key. If not specified, the `id` attribute of the tag is used.	R
cacheTimeout	No	String, long	The time, in milliseconds, for which the cached Content is valid. If more than this amount of time has passed since the Content was cached, the cached Content will be cleared, retrieved, and placed back into the cache. Use -1 for no-timeout (always use the cached Content). Default = -1.	R

Table 3-10 <pz:contentSelector> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
cacheScope	No	String	Specifies the lifecycle scope of the content cache. Similar to `<jsp:useBean>`. Possible values: ■ `application`. Any JSP page in the web application that any customer requests can access the cache. ■ `session` (the default). Any JSP in the web application that the current customer requests can access the cache. ■ `page`. Only the current JSP that any customer requests can access the cache. ■ `request`. Only the current user request can access the cache. If a customer re-requests the page, the content selector re-runs the query and recreates the cache.	R

Specify a Value for contentHome

The content selector tag must use the `contentHome` attribute to specify the JNDI home of the content management system. If you use the reference content management system or a third-party integration, you can use a scriptlet to refer to the default content home. Because the scriptlet uses the `ContentHelper` class, you must first use the following tag to import the class into the JSP:

```
<%@ page import="com.bea.p13n.content.ContentHelper"%>
```

Then, when you use the content selector tag, specify the `contentHome` as follows:

```
<pz:contentSelector
contentHome="<%=ContentHelper.DEF_DOCUMENT_MANAGER_HOME %>"
... />
```

If you create your own content management system, you must specify the JNDI home for your system instead of using the ContentHelper scriptlet. In addition, if your content management system provides a JNDI home, you can specify that one instead of using the ContentHelper scriptlet.

Example

Listing 3-10 shows how to use `<pz:contentSelector>`.

Listing 3-11 Using `<pz:contentSelector>`

```
<%@ page import="com.bea.p13n.content.ContentHelper" %>
<%@ taglib uri="es.tld" prefix="es" %>
<%@ taglib uri="cm.tld" prefix="cm" %>
<%@ taglib uri="pz.tld" prefix="pz" %>
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getProfile profileKey="bob"
profileId="myProfile" scope="session"/>
<pz:contentSelector rule="PremierCustomerSpotlight"
contentHome="<%=ContentHelper.DEF_DOCUMENT_MANAGER_HOME %>"
id="docs" />
<ul>
   <es:forEachInArray array="<%=docs%>" id="aDoc"
   type="com.bea.p13n.content.Content">
      <li>The document title is: <cm:printproperty id="aDoc"
      name="Title" encode="html" />
   </es:forEachInArray>
</ul>
```

Note: The `sortBy` attribute, when used in conjunction with the `max` attribute, works differently for explicit (system-defined) and implicit (user-defined) attributes. If you sort on explicit attributes (`identifier, mimeType, size, version, author, creationDate, modifiedBy, modifiedDate, lockedBy, description,` or `comments`) the sort is done on the database; therefore if you combine `max="10"` and `sortBy`, the system will perform the sort and then get the first 10 items. If you sort on implicit attributes, the sort is done *after* the max have been selected.

<pz:div>

Tag Library	`pz.tld`
Import Statement	`<%@ taglib uri="pz.tld" prefix="pz" %>`
Classes Implemented	DivTag DivTagExtraInfo

The `<pz:div>` tag allows a piece of content to be conditionally included as a result of a classifier rule being executed by the rules engine. If the user's profile matches the classification specified in the E-Business Control Center, then the conditional content is included. This tag has a begin tag, a body, and an end tag. The tag returns a list of `Classification` objects that the user belongs to.

Tag Attributes

Table 3-11 describes the `<pz:div>` tag attributes.

Table 3-11 <pz:div> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
rule	Yes	String	The name of the classifier rule in the customer segment definitions created in the E-Business Control Center.	R
id	No	String	A collection that contains the Classification objects that apply to the user for the given classifier rule.	C

Example

```
<%@ taglib uri="pz.tld" prefix="pz" %>
<%@ taglib uri="um.tld" prefix="um" %>

<um:getProfile profileKey="bob"
profileId="myProfile" scope="session"/>
```

```
<pz:div  id=”classifications” rule="PremierCustomer">


<%
//if the user is classified as a Premier Customer, a list with one
entry should be returned//
   java.util.Iterator iterator=classifications.iterator();
   while (iterator.hasNext())
   {
   com.bea.p13n.user. Classification
classification=(Classification) iterator.next();
  out.println (classification.getName());
   }
%>

   <p>Please check out our new Premier Customer bonus program.<p>
</pz:div>
```

Placeholders

The placeholder tag is a named location on a JSP. You use the E-Business Control Center to define the behavior of a placeholder.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<ph:placeholder>

Tag Library	`ph.tld`
Import Statement	`<%@ taglib uri="ph.tld" prefix="ph" %>>`
Classes Implemented	PlaceholderTag

The `<ph:placeholder>` tag implements a placeholder, which describes the behavior for a location on a JSP page. You use the E-Business Control Center to define a placeholder.

Multiple placeholder tags can refer to the same placeholder. Each instance of a placeholder tag invokes its placeholder definition separately. If the placeholder definition specifies multiple queries, each placeholder tag instance can display different ads, even though each instance shares the same definition.

When WebLogic Portal receives a request for a JSP that contains an ad placeholder, the placeholder tag contacts the Ad Service, a session EJB that invokes business logic to determine which ad to display.

For information on a related tag, see `<ad:adTarget>`.

How the Tag Maps to the Content Management Service Provider Interface

The following table shows how the tag accesses methods in the content management service provider interface (SPI). The table also shows the sequence of calls.

EJB/Data Object	SPI
1. `PlaceholderService.getContent()`	5. `DocumentProvider.findDocumentMetadata()`
2. `AdBucketService.getContent()`	6. `DocumentMetadataDef.getID()`
3. `AdService.getContent()`	7. `DocumentMetadataDef.getName()` ... (other `get*` methods)
4. `DocumentManager.getContent()`	8. `DocumentMetadataDef.getAttributeNames()`
10. `Document.getProperty()` ... (other `get*` methods)	9. `DocumentMetadataDef.getAttribute()`
11. `Document.getContent()`	13. `DocumentProvider.getDocument()`
12. `DocumentManager.getContentBlock()`	14. `DocumentDef.openStream()`

Tag Attributes

Table 3-12 describes the `<ph:placeholder>` tag attributes.

Table 3-12 <ph:placeholder> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
name	Yes	String	A string that refers to a placeholder definition.	R

Table 3-12 <ph:placeholder> Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
height	No	int	Specifies the height (in pixels) that the placeholder uses when generating the HTML that the browser requires to display a document. The placeholder uses this value only for content types to which display dimensions apply and only if other attributes have not already defined dimensions for a given document. If you do not specify this value and other attributes have not already been defined, the browser behavior determines the height of the document.	R
width	No	int	Specifies the width (in pixels) that the placeholder uses when generating the HTML that the browser requires to display a document. The placeholder uses this value only for content types to which display dimensions apply and only if other attributes have not already defined dimensions for a given document. If you do not specify this value and other attributes have not already been defined, the browser behavior determines the height of the document.	R

Example

Listing 3-12 displays the ad specified in the `MainPageBanner` placeholder.

Listing 3-12 Using `<ph:placeholder>`

```
<%@ taglib uri="ph.tld" prefix="ph" %>
.
.
```

```
.
<ph:placeholder name="/placeholders/MainPageBanner.pla"/>
```

Property Sets

The Property Set tags allow access to the list of available properties and property sets. Property sets are manipulated through the E-Business Control Center.

All Property Sets tags send results to the same file. If you are checking for results, include this import directive at the top of the page:

```
<%@ page
import="com.bea.p13n.property.servlets.jsp.taglib.PropertySetTag-
   Constants" %>
```

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<ps:getPropertyNames>

Tag Library	`ps.tld`
Import Statement	`<%@ taglib uri="ps.tld" prefix="ps" %>`
Classes Implemented	GetPropertyNamesTag GetPropertyNamesExtraInfo

The `<ps:getPropertyNames>` tag is used to get a list of property names given a property set.

Tag Attributes

Table 3-13 describes the `<ps:getPropertyNames>` tag attributes.

Table 3-13 <ps:getPropertyNames>

Tag Attribute	Required	Type	Description	R/C
propertySetName	Yes	String	The name of the property set to add the new search.	R
propertySetType	Yes	String	Type of property set to search.	R
id	Yes	String	The `id` of the variable to hold the list of property names, as a String array.	C
result	no	String	The identifier of an Integer variable that will be created and initialized with the result of the operation. Possible values: *Query is successful*: `PropertySetTagConstants.PROPERTY_SEARCH_OK` *Problem getting the list of property names*: `PropertySetTagConstants.PROPERTY_SEARCH_FAILED` *Property set named by* `propertySetName` *and* `propertySetType` *could not be found*: `PropertySetTagConstants.INVALID_PROPERTY_SET`	C

Example

Listing 3-13 shows how to use `<ps:getPropertyNames>`.

Listing 3-13 Using <ps:getPropertyNames>

```
<%@ taglib uri="ps.tld" prefix="ps" %>
<%@ page import=
"com.bea.p13n.property.servlets.jsp.taglib.PropertySetTagConstant
s"
%>

<% String myPropertySet="Demographics"; %>

<p> ------- <b>ps:getPropertyNames</b> -------------

<br>

<ps:getPropertyNames propertySetName="<%= myPropertySet %>"

propertySetType="USER" id="propertyNames" result="myResult"/>

<% for (int i=0; i<propertyNames.length; i++)

out.println(propertyNames[i] + "<br>");

%>
```

<ps:getPropertySetNames>

Tag Library	`ps.tld`
Import Statement	`<%@ taglib uri="ps.tld" prefix="ps" %>`
Classes Implemented	GetPropertySetNamesTag GetPropertySetNamesExtraInfo

The `<ps:getPropertySetNames>` tag (Table 3-14) is used to get a list of property sets given a property set type.

Tag Attributes

Table 3-14 describes the `<ps:getPropertySetNames>` tag attributes.

Table 3-14 <ps:getPropertySetNames>

Tag Attribute	Required	Type	Description	R/C
propertySetType	Yes	String	Type of property set to search.	R
id	Yes	String	The identifier of the variable to hold the list of property names, as a String array.	C
result	No	String	The identifier of an Integer variable that will be created and initialized with the result of the operation. Possible values: *Query is successful*: `PropertySetTagConstants.PROPERTY_SET_SEARCH_OK` *Problem getting the list of property names*: `PropertySetTagConstants.PROPERTY_SET_SEARCH_FAILED` *Property set named by* `propertySetName` *and* `propertySetType` *could not be found*: `PropertySetTagConstants.INVALID_PROPERTY_SET`	C

Example

Listing 3-14 shows how to use `<ps:getPropertySetNames>`.

Listing 3-14 Using `<ps:getPropertySetNames>`

```
<%@ taglib uri="ps.tld" prefix="ps" %>
.
.
.
<ps:getPropertySetNames propertySetType="USER"
id="userPropertySets" result="myResult" />
```

<ps:getRestrictedPropertyValues>

Tag Library	`ps.tld`
Import Statement	`<%@ taglib uri="ps.tld" prefix="ps" %>`
Classes Implemented	GetRestrictedPropertyValuesTag GetRestrictedPropertyValuesExtraInfo

The `<ps:getRestrictedPropertyValues>` tag returns a list of restricted values for a specific property definition, converted into Strings. These values will be returned as an array of Strings.

Tag Attributes

Table 3-15 describes the `<ps:getRestrictedPropertyValues>` tag attributes.

Table 3-15 <ps:getRestrictedPropertyValues>

Tag Attribute	Required	Type	Description	R/C
propertySetName	Yes	String	The name of the property set containing the property.	R
propertySetType	Yes	String	Type of property set containing the property.	R
propertyName	Yes	String	The name of the property to inspect.	R
id	Yes	String	The identifier of the variable to hold the list of property names, as a String array.	C

Table 3-15 <ps:getRestrictedPropertyValues> (Continued)

Tag Attribute	Required	Type	Description	R/C
result	No	String	The identifier of an Integer variable that will be created and initialized with the result of the operation. Possible values: *Query is successful*: `PropertySetTagConstants. PROPERTY_SEARCH_OK` *Problem accessing the property*: `PropertySetTagConstants. PROPERTY_SEARCH_FAILED` *Property set named by* `propertySetName` *and* `propertySetType` *could not be found*: `PropertySetTagConstants. INVALID_PROPERTY_SET` The requested property is not restricted : `PropertySetTagConstants. PROPERTY_NOT_RESTRICTED`	C

Example

Listing 3-15 shows how to use `<ps:getRestrictedPropertyValues>`.

Listing 3-15 Using `<ps:getRestrictedPropertyValues>`

```
<%@ taglib uri="ps.tld" prefix="ps" %>

<%@ page import=
"com.bea.p13n.property.servlets.jsp.taglib.PropertySetTagConstant
s"
%>

<p> ---- <b>ps:getRestrictedPropertyValues</b> -----
<br>Possible values for PreferredLanguage:
```

```
<ps:getRestrictedPropertyValues propertySetName="Demographics"
    propertySetType="USER" propertyName="PreferredLanguage"
    id="values" result="myResult"/>

<ul>
<% if (myResult.intValue() ==
PropertySetTagConstants.PROPERTY_SEARCH_OK)
  {
       for ( int i=0; i<values.length; i++ ) {
              %><li><%=values[i]%>
<% }
  }
%>
</ul>
```

User Management: Profile Management Tags

User Management tags allow access to user and group profile information, as well as operations such as creating and deleting users and groups, and managing user-group relationships.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<um:getProfile>

Tag Library

`um.tld`

Import Statement

```
<%@ taglib uri="um.tld" prefix="um" %>
```

All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:

```
<%@ page import="com.bea.p13n.usermgmt.
   servlets.jsp.taglib.UserManagementTag-
    Constants" %>
```

Classes Implemented

GetProfileTag

GetProfileExtraInfo

The `<um:getProfile>` tag (Table 3-16) retrieves the profile corresponding to the provided profile key and profile type. The tag has no enclosed body. The retrieved profile can be treated as a `com.bea.p13n.usermgmt.profile.ProfileWrapper`. Along with the profile key and profile, an explicit successor key and successor type can be specified, as specified by the `profileType` attribute. This successor will then be used, along with the retrieved profile, in subsequent invocations of the `<um:getProperty>` tag to ensure property inheritance from the successor. If no successor is retrieved, standard `ConfigurableEntity` successor search patterns will apply to retrieved properties.

Tag Attributes

Table 3-16 describes the `<um:getProfile>` tag attributes.

Table 3-16 <um:getProfile> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
profileKey	Yes	String	A unique identifier that can be used to retrieve the profile which is sought. Example: “<%=username%>”	R

Table 3-16 <um:getProfile> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
successorKey	No	String	A unique identifier that can be used to retrieve the profile successor. Example: “<%=defaultGroup%>”	R
scope	No	String	The HTTP scope of the retrieved profile. Pass "`request`" or "`session`" as the values. Defaults to *`session`*.	C
groupOnly	No	String	Specifies to retrieve a group profile named by the `profileKey`, rather than a user profile. No successor will be retrieved when this value is `true`. Defaults to `false`.	C
profileId	No	String	A variable name from which the retrieved profile is available for the duration of the JSP’s page scope.	C
successorId	No	String	A variable name from which the retrieved successor is available for the duration of the JSP’s page scope.	C

result	No	String	A variable name from which the result of the operation is available.
			Possible values:
			Success: `UserManagementTagConstants.GET_PROFILE_OK`
			Error encountered: `UserManagementTagConstants.GET_PROFILE_FAILED`
			`UserManagementTagConstants.NO_SUCH_PROFILE`
			`UserManagementTagConstants.NO_SUCH_SUCCESSOR`
				C

Example 1

Listing 3-16 shows a profile being retrieved with no successor specified and an explicitly-supplied *session* scope.

Listing 3-16 Using `<um:getProfile>`; Example 1

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getProfile profileKey="bob" profileType="AcmeUser"
profileId="myProfile" scope="session"/>
```

Example 2

Listing 3-17 shows a default user profile type being retrieved with a default successor type and an explicitly-supplied *request* scope.

Listing 3-17 Using `<um:getProfile>`; Example 2

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getProfile profileKey="bob" successorKey="engineering"
scope="request"/>
```

<um:getProperty>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.taglib.UserManagementTag-Constants" %>`

Classes Implemented	GetPropertyTag GetPropertyExtraInfo

The `<um:getProperty>` tag retrieves the property value for a specified property set-property name pair. The tag has no enclosed body. The value returned is an `Object`. In typical cases, this tag is used after the `<um:getProfile>` tag is invoked to retrieve a profile for session use. The property to be retrieved is retrieved from the session profile. If the `<um:getProfile>` tag has not been used upon invoking the `<um:getProperty>` tag, the specified property value is retrieved from the Anonymous User Profile.

Tag Attributes

Table 3-17 describes the `<um:getProperty>` tag attributes.

Table 3-17 <um:getProperty> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
propertySet	No	String	The Property Set from which the property's value is to be retrieved. Example: "Demographics" **Note:** If no property set is provided, the property is retrieved from the profile's default (unscoped) properties.	R

Table 3-17 <um:getProperty> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
propertyName	Yes	String	The name of the property to be retrieved. Example: “Date_of_Birth”	R
id	No	String	If the `id` attribute is supplied, the value of the retrieved property will be available in the variable name to which `id` is assigned. Otherwise, the value of the property is inlined.	C

Example 1

Listing 3-18 shows how to use `<um:getProperty>`.

Listing 3-18 Using `<um:getProperty>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getProperty id="myBirthDate" propertySet="Demographics"
propertyName="Date_of_Birth"/>
My birthday is <%=myBirthDate%>.
```

<um:getPropertyAsString>

Tag Library um.tld

Import Statement

```
<%@ taglib uri="um.tld" prefix="um" %>
```

All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:

```
<%@ page import="com.bea.p13n.usermgmt.
   servlets.jsp.taglib.UserManagementTag-
    Constants" %>
```

Classes Implemented

GetPropertyAsStringTag

GetPropertyAsStringExtraInfo

The <um:getPropertyAsString> tag works exactly like the <um:getProperty> tag above, but ensures that the retrieved property value is a String. The following example shows a multi-valued property which returns a Collection, but presents a list of favorite colors.

Tag Attributes

Table 3-18 describes the <um:getPropertyAsString> tag attributes.

Table 3-18 <um:getPropertyAsString> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
propertySet	No	String	The Property Set from which the property's value is to be retrieved. Example: "Demographics" **Note:** If no property set is provided, the property is retrieved from the profile's default (unscoped) properties.	R
propertyName	Yes	String	The name of the property to be retrieved. Example: "Date_of_Birth"	R

Table 3-18 <um:getPropertyAsString> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
id	No	String	If the `id` attribute is supplied, the value of the retrieved property will be available in the variable name to which `id` is assigned. Otherwise, the value of the property is inlined.	C

Example

Listing 3-19 shows how to use `<um:getPropertyAsString>`.

Listing 3-19 Using `<um:getPropertyAsString>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getPropertyAsString id="myBirthDate"
propertySet=”Demographics” propertyName=”Date_of_Birth”/>
My birthday is <%=myBirthDate%>.
```

<um:removeProperty>

Tag Library

`um.tld`

Import Statement

`<%@ taglib uri="um.tld" prefix="um" %>`

All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:

```
<%@ page import="com.bea.p13n.usermgmt.
   servlets.jsp.taglib.UserManagementTag-
    Constants" %>
```

Classes Implemented

RemovePropertyTag

The `<um:removeProperty>` tag (Table 3-19) removes the specified property from the current session's profile or from the Anonymous User Profile. The tag has no enclosed body. Subsequent calls to `<um:getProperty>` for a removed property would result in the default value for the property as prescribed by the property set, or from the Profile's successor.

Tag Attributes

Table 3-19 describes the `<um:removeProperty>` tag attributes.

Table 3-19 <um:removeProperty> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
propertySet	No	String	The Property Set from which the property's value is to be removed. Example: "Demographics" **Note:** The property is removed from the profile's default (unscoped) properties if no property set is provided.	R
propertyName	Yes	String	The name of the property to be removed. Example: "Income_Range"	R

Example

Listing 3-20 shows how to use `<um:removeProperty>` to remove a property.

Listing 3-20 Using `<um:removeProperty>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:removeProperty propertySet="<%=thePropertySet%>"
propertyName="<%=thePropertyName%>"/>
```

<um:setProperty>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.taglib.UserManagementTag-Constants" %>`

Classes Implemented	SetPropertyTag ResultTagExtraInfo

The `<um:setProperty>` tag updates a property value for either the session's current profile, or for the Anonymous User Profile. This tag has no enclosed body.

Tag Attributes

Table 3-20 describes the `<um:setProperty>` tag attributes.

Table 3-20 <um:setProperty> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
propertySet	No	String	The Property Set in which the property's value is to be set. Example: "Demographics" **Note:** The property is set for the profile's default (unscoped) properties if no property set is provided.	R
propertyName	Yes	String	The name of the property to be set. Example: "Gender"	R
value	Yes	Object	The new property value.	R
result	No	String	The name of an Integer object to which the result of the set property operation is assigned. *Success:* `UserManagementTagConstants.SET_PROPERTY_OK` *Error encountered:* `UserManagementTagConstants.SET_PROPERTY_FAILED`	C

Example

Listing 3-21 shows how to use `<um:setProperty>`.

Listing 3-21 Using `<um:setProperty>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<% String myGender = request.getParameter("gender"); %>
<um:setProperty propertySet="Demographics" propertyName="Gender"
value="<%=myGender%>"/>
```

User Management: Group-User Management Tags

User Management tags allow access to user and group profile information, as well as operations such as creating and deleting users and groups, and managing user-group relationships.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<um:addGroupToGroup>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt. servlets.jsp.tags.UserManagementTag- Constants" %>`

Classes Implemented	AddGroupToGroupTag ResultTagExtraInfo

The `<um:addGroupToGroup>` tag (Table 3-21) adds the group corresponding to the provided `childGroupName` to the group corresponding to the provided `groupName`. Since a group can only have one parent, any previous database records which reflect the group belonging to another parent will be destroyed. Both the parent group and the child group must previously exist for proper tag behavior. The tag has no enclosed body.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-21 describes the `<um:addGroupToGroup>` tag attributes.

Table 3-21 <um:addGroupToGroup>

Tag Attribute	Required	Type	Description	R/C
childGroupName	Yes	String	The name of the child group. Example: “<%=childGroupName%>”	R
parentGroupName	No	String	The name of the parent group. Example: “<%=parentGroupName%>”	R
result	Yes	String	The name of an Integer variable to which the result of the add group to group operation is assigned. Possible values: *Success*: `UserManagementTagConstants.ADD_GROUP_OK` *Error encountered*: `UserManagementTagConstants.ADD_GROUP_FAILED`	C

Example

Listing 3-22 shows how to use `<um:addGroupToGroup>` to add a new group of users to an existing group of users.

Listing 3-22 Using `<um:addGroupToGroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
```

```
.
.
<um:addGroupToGroup childGroupName=“<%=childGroupName%>”
parentGroupName=”<%=parentGroupName%>” result=”result”/>
```

<um:addUserToGroup>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>`

Classes Implemented	AddUserToGroupTag ResultTagExtraInfo

The `<um:addUserToGroup>` tag adds the user corresponding to the provided `username` to the group corresponding to the provided `groupName`. Both the specified user and the specified group must previously exist for proper tag behavior. The tag has no enclosed body.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-22 describes the `<um:addUserToGroup>` tag attributes.

Table 3-22 <um:addUserToGroup>

Tag Attribute	Required	Type	Description	R/C
username	Yes	String	The name of the user to be added to the group. Example: "<%=username%>"	R
groupName	Yes	String	The name of the group to which the user is being added. Example: "<%=groupName%>"	R
result	Yes	String	The name of an Integer variable to which the result of the add user to group operation is assigned. Possible values: *Success:* `UserManagementTagConstants.ADD_USER_OK` *Error encountered:* `UserManagementTagConstants.ADD_USER_FAILED`	C

Example

Listing 3-23 shows how to use `<um:addUserToGroup>` to add a new user to an existing group.

Listing 3-23 Using `<um:addUserToGroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:addUserToGroup username="<%=username%>"
groupName="<%=groupName%>" result="result"/>
```

<um:createGroup>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>`

Classes Implemented	CreateGroupTag CreateGroupExtraInfo

The `<um:createGroup>` tag (Table 3-23) creates a new group in the realm, and a corresponding group profile in the personalization database. This tag has no enclosed body.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-23 describes the `<um:createGroup>` tag attributes.

Table 3-23 <um:createGroup> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
groupName	Yes	String	The name of the new group. Example: “<%=groupName%>”	R
id	No	String	A variable name to which the created Group object is available for the duration of the page’s scope.	C

Table 3-23 <um:createGroup> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
parentName	No	String	The name of the group to set as the parent of the new group.	R
result	Yes	String	The name of an Integer variable to which the result of the create group operation is assigned. Possible Values: *Success:* `UserManagementTagConstants.CREATE_GROUP_OK` *Error encountered:* `UserManagementTagConstants.CREATE_GROUP_FAILED` *A group with the specified group name already exists:* `UserManagementTagConstants.GROUP_EXISTS`	C

Example

Listing 3-24 shows how to use `<um:creategroup>` to create a new group.

Listing 3-24 Using `<um:creategroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:creategroup groupName="<%=groupName%>" result="result"/>
```

<um:createUser>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTagConstants" %>`

Classes Implemented	CreateUserTag CreateUserExtraInfo

The `<um:createUser>` tag creates a new user profile. This tag has no enclosed body. Although classified as a Group-User management tag, this tag can be used in conjunction with run-time activities, in that it will persist any properties associated with a current Anonymous User Profile if specified.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-24 describes the `<um:createUser>` tag attributes.

Table 3-24 <um:createUser> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
username	Yes	String	The name of the new user. Example: “<%=username%>”	R
password	Yes	String	The password for the new user. Example: “<%=password%>”	R

Table 3-24 <um:createUser> Tag Attributes

Tag Attribute	Required	Type	Description	R/C

profileType	No	String	Specifies the extended type of user (for example, WLCS_Customer) to create a user of that type.
			You can set a default profile type for each Web application by setting a context parameter in `web.xml` for DEFAULT_USER_PROFILE_TYPE. For example:
			`<context-param>`
			`<param-name>DEFAULT_USER_PROFILE_TYPE</param-name>`
			`<param-value>WLCS_Customer</param-value>`
			`</context-param>`
				R

saveAnonymous	No	String	Whether to persist current anonymous user profile attributes in the newly-created user's profile. Defaults to `false`. Example: "`false`"	R
id	No	String	A variable name to which the created User object is available for the duration of the page's scope.	C
result	Yes	String	The name of an Integer variable to which the result of the create user operation is assigned. Possible values: *Success:* `UserManagementTagConstants.CREATE_USER_OK` *Error encountered:* `UserManagementTagConstants.CREATE_USER_FAILED` *A user with the specified username already exists:* `UserManagementTagConstants.USER_EXISTS`	C

Example

Listing 3-25 shows how to use `<um:createUser>` to create a new user.

Listing 3-25 Using `<um:createUser>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
<um:createUser username="<%=username%>" password="<%=password"%>
result="result"/>
```

<um:getChildGroupNames>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>`

Classes Implemented	GetChildGroupNamesTag GetChildGroupNamesExtraInfo

The `<um:getChildGroupNames>` tag returns the names of any groups that are children of the given group.

Tag Attributes

Table 3-25 describes the `<um:getChildGroupNames>` tag attributes.

Table 3-25 <um:getChildGroupNames>

Tag Attribute	Required	Type	Description	R/C
groupName	Yes	String	The name of the group to search for child groups.	R
id	Yes	String	The name of the identifier which will be assigned the String array of child group names.	C

Example

Listing 3-26 shows how to use `<um:getChildGroupNames>` to retrieve the names of a child group for the group `SomeGroup`.

Listing 3-26 Using `<um:getChildGroupNames>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getChildGroupNames groupName="SomeGroup"
id="childrenOfSomeGroup" />
```

<um:getGroupNamesForUser>

Tag Library `um.tld`

Import Statement `<%@ taglib uri="um.tld" prefix="um" %>`

All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:

```
<%@ page import="com.bea.p13n.usermgmt.
   servlets.jsp.tags.UserManagementTag-
   Constants" %>
```

Classes Implemented GetGroupNamesForUserTag

GetGroupNamesForUserExtraInfo

The `<um:getGroupNamesForUser>` tag retrieves a `String` array that contains the group names corresponding to groups to which the provided user immediately belongs. This tag has no enclosed body.

Tag Attributes

Table 3-26 describes the `<um:getGroupNamesForUser>` tag attributes.

Table 3-26 <um:getGroupNamesForUser> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
username	Yes	String	The name of the user whose matching groups are sought. Example: “<%=username%>”	R
id	Yes	String	A variable name to which the resultant group names are assigned. Example: “myGroups”	C

Example

Listing 3-27 shows how to use `<um:getGroupNamesForUser>` to retrieve a group name to apply to a user.

Listing 3-27 Using `<um:getGroupNamesForUser>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getGroupNamesForUser username="<%=username%>" id="myGroups"/>
```

<um:getParentGroupName>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>`

Classes Implemented	GetParentGroupNameTag GetParentGroupNameExtraInfo

The `<um:getParentGroupName>` tag retrieves the name of the parent of the group associated with the provided `groupName`. The information is taken from the realm. This tag has no enclosed body.

Tag Attributes

Table 3-27 describes the `<um:getParentGroupName>` tag attributes.

Table 3-27 <um:getParentGroupName>

Tag Attribute	Required	Type	Description	R/C
groupName	Yes	String	The name of the group whose parent group name is sought. Example: “<%=groupName%>”	R
id	Yes	String	A variable name to which the name of the parent is available for the duration of the page’s scope. Example: “parentGroupName”	C

Example

Listing 3-28 shows how to use `<um:getParentGroupName>` to retrieve a parent group name.

Listing 3-28 Using `<um:getParentGroupName>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getParentGroupName groupName="<%=groupName%>"
id="parentGroupName"/>
```

<um:getTopLevelGroups>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>`

Classes Implemented	GetTopLevelGroupsTag GetTopLevelGroupsExtraInfo

The `<um:getTopLevelGroups>` tag retrieves an array of group names, each of which has no parent group. The information is taken from the realm. This tag has no enclosed body.

Tag Attributes

Table 3-28 describes the `<um:getTopLevelGroups>` tag attributes.

Table 3-28 <um:getTopLevelGroups>

Tag Attribute	Required	Type	Description	R/C
id	Yes	String	A variable name to which the top-level `Group` objects are available for the duration of the page's scope. Example: "topLevelGroups"	C

Example

Listing 3-29 shows how to use `<um:getTopLevelGroups>`.

Listing 3-29 Using `<um:getTopLevelGroups>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getTopLevelGroups id=”topLevelGroups”/>
```

<um:getUsernames>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTagConstants" %>`

Classes Implemented	GetUsernamesTag GetUsernamesExtraInfo

The `<um:getUsernames>` tag retrieves a `String` array that contains the usernames matching the provided search expression. The search expression supports only the asterisk (*) wildcard character, and is case insensitive. As many asterisks as desired may be used in the search expression. This tag has no enclosed body.

Tag Attributes

Table 3-29 describes the `<um:getUsernames>` tag attributes.

Table 3-29 <um:getUsernames> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
searchExp	No	String	The search expression to apply to the user name search. Defaults to '*' Example: "t*"	R
userLimit	No	String (representing an Integer)	The maximum number of users to be returned from the search. (String which has a particular `Integer.valueOf.`) Defaults to 100. If user count exceeds userLimit, the length of the array in `id` is truncated to the length of userLimit. Example: "500"	R
id	Yes	String	A variable name to which the resultant user names are assigned. Example: "myUsers"	C
result	No	String	The name of an Integer variable to which the result of the getUsernames operation is assigned. Possible values: *Success:* UserManagerTagConstants.USER_SEARCH_OK *General error* (such as a database connection failure that occurred during the search): *UserManagerTagConstants*.USER_SEARCH_FAILED **Note:** If no users match the search criteria, then the result will not be equal to UserManagerTagConstants.USER_SEARCH_FAILED, but the length of the array returned in "id" will be zero.	C

Note: The USER_SEARCH_FAILED value is returned only when a general error occurs while searching for the user, such as a database connection failure. If no user matches the search criteria, the result will not be equal to UserManagementTagConstants.USER_SEARCH_FAILED, but the length returned by the array in id will be zero.

Example

Listing 3-30 shows how to use <um:getUsernames> to retrieve a username.

Listing 3-30 Using <um:getUsernames>

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getUsernames userLimit="500" searchExp="t*" id="myUsers"/>
<%System.out.println("I found " + myUsers.length + " users.");%>
```

<um:getUsernamesForGroup>

Tag Library	um.tld
Import Statement	<%@ taglib uri="um.tld" prefix="um" %> All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page: <%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>
Classes Implemented	GetUsernamesForGroupTag GetUsernamesForGroupExtraInfo

The `<um:getUsernamesForGroup>` tag retrieves a `String` array that contains the usernames matching the provided search expression and correspond to members of the provided group. The search expression supports only the asterisk (*) wildcard character, and is case insensitive. As many asterisks as desired may be used in the search expression. This tag has no enclosed body.

Tag Attributes

Table 3-30 describes the `<um:getUsernamesForGroup>` tag attributes.

Table 3-30 <um:getUsernamesForGroup> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
searchExp	No	String	The search expression to apply to the user name search. Defaults to " *". Example: `"t*"`	R
groupName	Yes	String	The name of the group whose matching members are sought. Example: “engineering”	R
userLimit	No	String (representing an Integer)	The maximum number of users to be returned from the search. (String which has a particular `Integer.valueOf.`) Defaults to 100. If user count exceeds userLimit, the length of the array in id is truncated to the length of userLimit. Example: “500”	R
id	Yes	String	A variable name to which the resultant user names are assigned. Example: “myUsers”	C

Table 3-30 <um:getUsernamesForGroup> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
result	No	String	The name of an Integer variable to which the result of the get usernames for group operation is assigned. Possible values: *Success:* `UserManagementTagConstants.USER_SEARCH_OK` *General error:* `UserManagementTagConstants.USER_SEARCH_FAILED`	C

Note: The `USER_SEARCH_FAILED` value is returned only when a general error occurs while searching for the user, such as a database connection failure. If no user matches the search criteria, the result will not be equal to `UserManagementTagConstants.USER_SEARCH_FAILED`, but the length returned by the array in `id` will be zero.

Example

Listing 3-31 shows how to use `<um:getUsernamesForGroup>` to retrieve usernames that match the provided search expression and correspond to members of the provided group.

Listing 3-31 Using `<um:getUsernamesForGroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:getUsernamesForGroup groupName="engineering" userLimit="500"
searchExp="t*" id="myUsers"/>
<%System.out.println("I found " + myUsers.length + " users in my
group.");%>
```

<um:removeGroup>

Tag Library `um.tld`

Import Statement

```
<%@ taglib uri="um.tld" prefix="um" %>
```

All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:

```
<%@ page import="com.bea.p13n.usermgmt.
   servlets.jsp.tags.UserManagementTag-
   Constants" %>
```

Classes Implemented

RemoveGroupTag

ResultTagExtraInfo

The `<um:removeGroup>` tag removes the group corresponding to the provided `groupName`. This tag has no enclosed body.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-31 describes the `<um:removeGroup>` tag attributes.

Table 3-31 <um:removeGroup>

Tag Attribute	Required	Type	Description	R/C
groupName	Yes	String	The name of the group to be removed. Example: “<%=groupName%>”	R

Table 3-31 <um:removeGroup> (Continued)

Tag Attribute	Required	Type	Description	R/C
result	Yes	String	The name of an Integer variable to which the result of the remove group operation is assigned. Possible Values: *Success:* `UserManagementTagConstants.REMOVE_GROUP_OK` *Error encountered:* `UserManagementTagConstants.REMOVE_GROUP_FAILED`	C

Example

Listing 3-32 shows how to use `<um:removeGroup>` to remove a group that corresponds to the provided `groupName`.

Listing 3-32 Using `<um:removeGroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:removeGroup groupName=”<%=groupName%>” result=”result”/>
```

<um:removeGroupFromGroup>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTagConstants" %>`

Classes Implemented	RemoveGroupFromGroupTag ResultTagExtraInfo

The `<um:removeGroupFromGroup>` tag removes a child group from a parent group.

Tag Attributes

Table 3-32 describes the `<um:removeGroupFromGroup>` tag attributes.

Table 3-32 <um:removeGroupFromGroup>

Tag Attribute	Required	Type	Description	R/C
childGroupName	Yes	String	The name of the child group to remove from its parent.	R
parentGroupName	Yes	String	The name of the parent group from which the child group will be removed.	R

Table 3-32 <um:removeGroupFromGroup> (Continued)

Tag Attribute	Required	Type	Description	R/C
result	Yes	String	The name of an Integer variable to which the result of the remove group from group operation is assigned. Possible values: *Success:* `UserManagementTagConstants.REMOVE_GROUP_OK` *Failure:* `UserManagementTagConstants.REMOVE_GROUP_FAILED`	C

Example

Listing 3-33 shows how to use `<um:removeGroupFromGroup>` to remove a child group from a parent group.

Listing 3-33 Using `<um:removeGroupFromGroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:removeGroupFromGroup parentGroupName="SomeGroup"
childGroupName="ChildGroupToRemove" result="myResult" />
```

<um:removeUser>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTagConstants" %>`

Classes Implemented	RemoveUserTag ResultTagExtraInfo

The `<um:removeUser>` tag removes the user corresponding to the provided `username`. It can remove any type of extended user that has its profileType set in the database. This tag has no enclosed body.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-33 describes the `<um:removeUser>` tag attributes.

Table 3-33 <um:removeUser>

Tag Attribute	Required	Type	Description	R/C
username	Yes	String	The username of the user to be removed. Example: “<%=username%>”	R

Table 3-33 <um:removeUser> (Continued)

Tag Attribute	Required	Type	Description	R/C
result	Yes	String	The name of an Integer variable to which the result of the remove user operation is assigned. Possible values: *Success:* `UserManagementTagConstants.REMOVE_USER_OK` *Error encountered:* `UserManagementTagConstants.REMOVE_USER_FAILED`	C

Example

Listing 3-34 shows how to use `<um:removeUser>` to remove a user.

Listing 3-34 Using `<um:removeUser>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:removeUser username=”<%=username%>” result=”result”/>
```

<um:removeUserFromGroup>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.tags.UserManagementTag-Constants" %>`

Classes Implemented	RemoveUserFromGroupTag ResultTagExtraInfo

The `<um:removeUserFromGroup>` tag removes a user from a group.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`).

Tag Attributes

Table 3-34 describes the `<um:removeUserFromGroup>` tag attributes.

Table 3-34 <um:removeUserFromGroup>

Tag Attribute	Required	Type	Description	R/C
username	Yes	String	The username of the user to remove from the given group.	R
groupName	Yes	String	The name of the group from which the given user will be removed.	R

Table 3-34 <um:removeUserFromGroup> (Continued)

Tag Attribute	Required	Type	Description	R/C
result	Yes	String	The name of an Integer variable to which the result of the remove user from group operation is assigned. Possible values: *Success:* `UserManagementTagConstants.REMOVE_USER_OK` *Failure:* `UserManagementTagConstants.REMOVE_USER_FAILED`	C

Example

Listing 3-35 shows how to use `<um:removeUserFromGroup>` to remove a specified user from a specified group.

Listing 3-35 Using `<um:removeUserFromGroup>`

```
<%@ taglib uri="um.tld" prefix="um" %>
.
.
.
<um:removeUserFromGroup username="UserToRemove"
groupName="SomeGroup" result="myResult" />
```

User Management: Security Tags

User Management tags allow access to user and group profile information, as well as operations such as creating and deleting users and groups, and managing user-group relationships.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<um:login>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt. servlets.jsp.taglib.UserManagement- TagConstants" %>`

Classes Implemented	LoginTag ResultTagExtraInfo

The `<um:login>` tag provides weak authentication (username, password) against the current security realm, and sets the authenticated user as the current WebLogic user. This tag has no enclosed body.

Note: The login tag requires a `username` parameter and a `password` parameter to be present in the HTTP request.

Tag Attributes

Table 3-35 describes the `<um:login>` tag attributes.

Table 3-35 <um:login> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
result	Yes	String	The name of an Integer variable to which the result of the login operation is assigned. Possible values: *Success:* `UserManagementTagConstants.LOGIN_OK` *General error when performing authentication:* `UserManagementTagConstants.LOGIN_ERROR` *Authentication failed because of invalid username/password combination*: `UserManagementTagConstants.LOGIN_FAILED`	C

<um:logout>

Tag Library `um.tld`

Import Statement `<%@ taglib uri="um.tld" prefix="um" %>`

All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:

```
<%@ page import="com.bea.p13n.usermgmt.
   servlets.jsp.taglib.UserManagement-
   TagConstants" %>
```

Classes Implemented LogoutTag

ResultTagExtraInfo

The `<um:logout>` tag ends the current user's WebLogic Server session. This tag should be used in combination with the `<um:login>` tag.

Tag Attributes

Table 3-36 describes the `<um:logout>` tag attributes.

Table 3-36 <um:logout>

Tag Attribute	Required	Type	Description	R/C
No attributes				

<um:setPassword>

Tag Library	`um.tld`

Import Statement	`<%@ taglib uri="um.tld" prefix="um" %>`
	All User Management tags send results to the same file. If you are checking for results, include this import statement at the top of the page:
	`<%@ page import="com.bea.p13n.usermgmt.servlets.jsp.taglib.UserManagementTagConstants" %>`

Classes Implemented	SetPasswordTag ResultTagExtraInfo

The `<um:setPassword>` tag updates the password for the user corresponding to the provided username.

Note: This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl.ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm.RDBMSRealm`). In addition, the user object used by the current realm must implement `weblogic.security.acl.CredentialChanger`.

Tag Attributes

Table 3-37 describes the `<um:setPassword>` tag attributes.

Table 3-37 <um:setPassword> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
username	Yes	String	The username of the user whose password is to be changed.	R
password	Yes	String	The new user password.	R
result	Yes	String	The name of an Integer variable to which the result of the set password operation is assigned. Possible values: *Success:* `UserManagementTagConstants.SET_PASSWORD_OK` *Failure:* `UserManagementTagConstants.SET_PASSWORD_FAILED`	C

Personalization Utilities

The `<es:jsptaglib>` tag contains generic tags you can use to create JSP pages.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<es:convertSpecialChars>

Tag Library	`es.tld`
Import Statement	`<%@ taglib uri="es.tld" prefix="es" %>`
Classes Implemented	ConvertSpecialCharsTag

The `<es:convertSpecialChars>` tag converts characters which would normally signify special meaning to an HTML browser into characters which can be displayed as intended.

For example, the following sentence must be converted because it uses the “<“ and “>”characters, which signify tag opening and closing to the browser:

```
Enter <filename> here:
```

Tag Attributes

Table 3-38 describes the `<es:convertSpecialChars>` tag attributes.

Table 3-38 <es:convertSpecialChars>

Tag Attribute	Required	Type	Description	R/C
string	Yes	String	The string to be converted.	R

Example

Listing 3-36 allows a string containing a less-than sign to be rendered in HTML.

Listing 3-36 Using `<es:convertSpecialChars>`

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
<es:convertSpecialChars string="<thisString>"/>
```

<es:counter>

Tag Library	`es.tld`
Import Statement	`<%@ taglib uri="es.tld" prefix="es" %>`
Classes Implemented	CounterTag CounterTagExtraInfo

The `<es:counter>` tag is used to create a `for` loop.

Tag Attributes

Table 3-39 describes the `<es:counter>` tag attributes.

Table 3-39 <es:counter>

Tag Attribute	Required	Type	Description	R/C
type	No	String	The type of the counter. Possible values are `int` or `long`. Default is `int`.	R
id	Yes	String	A unique name for the variable.	R

Table 3-39 <es:counter> (Continued)

Tag Attribute	Required	Type	Description	R/C
minCount	Yes	Int	The start position for the loop.	R
maxCount	Yes	Int	The end position for the loop.	R

Example

Listing 3-37 shows how to use `<es:counter>`.

Listing 3-37 Using `<es:counter>`

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
<es:counter id="iterator" minCount="0" maxCount="10">
   <% System.out.println(iterator);%>
</es:counter>
```

<es:date>

Tag Library	`es.tld`
Import Statement	`<%@ taglib uri="es.tld" prefix="es" %>`
Classes Implemented	DateTag

The `<es:date>` tag is used to get a date- and time-formatted String based on the user's time zone preference.

Tag Attributes

Table 3-40 describes the `<es:date>` tag attributes.

Table 3-40 <es:date>

Tag Attribute	Required	Type	Description	R/C
timeZoneId	No	String	Defaults to the time zone on the server.	R
formatStr	No	String	A date and time format string that adheres to the java.text.SimpleDateFormat. The default value is *MM/dd/yyyy HH:mmss:z*.	R

Example

Listing 3-38 shows how <es:date> is used to retrieve a time and date.

Listing 3-38 Using <es:date>

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
<es:date formatStr="MMMM dd yyyy" timeZoneId="MST" />
```

<es:forEachInArray>

Tag Library	es.tld
Import Statement	<%@ taglib uri="es.tld" prefix="es" %>
Classes Implemented	ForEachInArrayTag ForEachInArrayTagExtraInfor

The <es:forEachInArray> tag is used to iterate over an array.

Tag Attributes

Table 3-41 describes the `<es:forEachInArray>` tag attributes.

Table 3-41 <es:forEachInArray>

Tag Attribute	Required	Type	Description	R/C
id	Yes	String	The variable for each value in the array.	R
type	Yes	String	The type of each value in the array.	R
array	Yes	Object `[]`	The array to iterate over.	R
counterId	No	String	The position in the array.	R

Example

Listing 3-39 shows how to use `<es:forEachInArray>` to iterate over an array.

Listing 3-39 Using `<es:forEachInArray>`

```
<es:forEachInArray id="item" array="<%=items%>" type="String"
counterId="i">
     <% System.out.println("items[" + i + "]: " + item);%>
</es:forEachInArray>
```

<es:isNull>

Tag Library	`es.tld`
Import Statement	`<%@ taglib uri="es.tld" prefix="es" %>`
Classes Implemented	IsNullTag

The <es:isNull> tag is used to check if a value is null. In the case of a String, the <es:isNull> tag is used to check if the String is null or has a value. An empty string will cause isNull to be false. (An empty string is not null.)

Tag Attributes

Table 3-42 describes the <es:isNull> tag attributes.

Table 3-42 <es:isNull> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
item	Yes	Object	The variable to evaluate.	R

Example

Listing 3-40 shows how to use <es:isNull> to check if a value is null.

Listing 3-40 Using <es:isNull>

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
<es:isNull item="<%=value%>">
     Error: the value is null.
</es:isNull>
```

<es:notNull>

Tag Library	es.tld
Import Statement	<%@ taglib uri="es.tld" prefix="es" %>
Classes Implemented	NotNullTag

The <es:notNull> tag is used to check if a value is not null. In the case of a String, the <es:notNull> tag is used to check if the String is not null or has a value. An empty string will cause notNull to be true. (An empty string is treated as a value.)

Tag Attributes

Table 3-43 describes the <es:notNull> tag attributes.

Table 3-43 <es:notNull>

Tag Attribute	Required	Type	Description	R/C
item	Yes	Object	The variable to evaluate.	R

Example

Listing 3-41 shows how to use <es:notNull> to check if a value is not null.

Listing 3-41 Using <es:notNull>

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
<es:notNull item="<%=value%>">
     The value is not null.
</es:notNull>
```

<es:transposeArray>

Tag Library	`es.tld`
Import Statement	`<%@ taglib uri="es.tld" prefix="es" %>`
Classes Implemented	TransposeArrayTag TransposeArrayTagExtraInfo

The `<es:transposeArray>` tag is used to transpose a standard [row][column] array to a [column][row] array.

Tag Attributes

Table 3-44 describes the `<es:transposeArray>` tag attributes.

Table 3-44 <es:transposeArray>

Tag Attribute	Required	Type	Description	R/C
id	Yes	String	The variable that holds the [c][r] array.	R
type	Yes	String	The type of variable in the [r][c] array, such as String.	R
array	Yes	Object`[][]`	The variable that holds the [r][c] array.	R

Example

Listing 3-42 shows how to use `<es:transposeArray>`.

Listing 3-42 Using `<es:transposeArray>`

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
```

```
<es:transposeArray id="byColumnRow" array="<%=byRowColumn%>"
type="String">
   ...
</es:transposeArray>
```

<es:uriContent>

Tag Library	`es.tld`
Import Statement	`<%@ taglib uri="es.tld" prefix="es" %>`
Classes Implemented	UriContentTag UriContentTagExtraInfo

The `<es:uriContent>` tag is used to pull content from a URL. It is best used for grabbing text-heavy pages.

Tag Attributes

Table 3-45 describes the `<es:uriContent>` tag attributes.

Table 3-45 <es:uriContent>

Tag Attribute	Required	Type	Description	R/C
id	Yes	String	The variable that holds the downloaded content of the URI.	R
uri	Yes	String	The fully qualified URI from which to get the content.	R

Example

Listing 3-43 shows how to use `<es:uriContent>` to pull content from a URL.

Listing 3-43 Using `<es:uriContent>`

```
<%@ taglib uri="es.tld" prefix="es" %>
.
.
.
<es:uriContent id="uriContent"
uri="http://www.beasys.com/index.html">
<%
   out.print(uriContent);
%>
</es:uriContent>
```

Note: If you combine HTML pages with relative URL's, you must fully qualify them to the correct host in each URL, or else images (on other resources) may not be retrieved properly by the browser.

WebLogic Utilities

The `<wl:jsptaglib>` tag library contains custom JSP extension tags which are supplied as a part of the WebLogic Server platform.

In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<wl:cache>

Tag Library	`wl.tld`
Import Statement	`<%@ taglib uri="weblogic.tld" prefix="wl" %>`
Classes Implemented	CacheTag CacheTagInfo

The `<wl:cache>` tag specifies that its contents do not necessarily need to be updated every time it is displayed.

Tag Attributes

Table 3-46 describes the `<wl:cache>` tag attributes.

Table 3-46 <wl:cache>

Tag Attribute	Required	Type	Description	R/C
timeout	No	Integer	Controls the time-to-live of the data, or how often the data must be updated independent of all other controls. This value is in seconds.	R

Table 3-46 <wl:cache>

Tag Attribute	Required	Type	Description	R/C
scope	No	String	Controls the time-to-live of the data, or how often the data must be updated independent of all other controls. This value is in seconds	C
name	No	String	Uniquely identifies this cache. If you do not specify a name a random name will be generated.	C
size	No	Integer	The maximum number of entries that can be in the cache. It defaults to an unlimited cache. It is only relevant for when there is an associated key.	R
vars	No	String	In addition to caching the transformed output of the cache, you can also cache calculated values within the block. These variables are specified exactly the same way as the cache keys. This type of caching is called Input caching.	C
key	No	String	Specifies a comma separated list of values accessible from the current page that the data depends on. These values act as additional keys into the cache.	C
async	No	String	If the async parameter is set to `true`, the cache will be updated asynchronously, if possible. The user that initiates the cache hit sees the old data.	C

<wl:process>

Tag Library	`wl.tld`
Import Statement	`<%@ taglib uri="weblogic.tld" prefix="wl" %>`
Classes Implemented	ProcessTag

The `<wl:process>` tag is used for query attribute-based flow control. By using a combination of the four attributes, you can selectively execute the statements between the `<wl:process>` and `</wl:process>` tags, as shown in Listing 3-44.

Tag Attributes

Table 3-47 describes the `<wl:process>` tag attributes.

Table 3-47 <wl:process>

Tag Attribute	Required	Type	Description	R/C
name	No	String	The name of a query attribute.	R
notName	No	String	The name of a query attribute.	R
value	No	String	The value of a query attribute.	R
notValue	No	String	The value of a query attribute.	R

Statements between the `<wl:process>` tags will be executed according to the matrix below:

Attribute	Value	notValue	Neither "value" nor "notValue"
name	Named attribute is equal to the value.	Named attribute does not equal the value.	Name attribute's value is not null.
not Name			notName attribute's value is null.

Example

Listing 3-44 shows how to use `<wl:process>`. It will execute if `lastBookRead` exists and the value of `lastBookRead` is `A Man in Full`.

Listing 3-44 Using `<wl:process>`

```
<%@ taglib uri="weblogic.tld" prefix="wl" %>
.
.
.
<wl:process name="lastBookRead" value="A Man in Full">
<!-- This section of code will be executed
   if lastBookRead exists and the value of lastBookRead is
   "A Man in Full" -->
</wl:process>
```

<wl:repeat>

Tag Library	`wl.tld`
Import Statement	`<%@ taglib uri="weblogic.tld" prefix="wl" %>`
Classes Implemented	RepeatTag RepeatTagInfo

The `<wl:repeat>` tag is used to iterate over a variety of Java objects, as specified in the set attribute.

Tag Attributes

Table 3-48 describes the `<wl:repeat>` tag attributes.

Table 3-48 <wl:repeat> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
set	No	Object	The set of objects that includes: ■ Enumerations ■ Iterators ■ Collections ■ Arrays ■ Vectors ■ Result Sets ■ Result Set MetaData ■ Hashtable keys	R
count	No	Int	Iterate over first “count” entries in the set.	R
id	No	String	Variable to contain current object being iterated over.	C
type	No	String	Type of object that results from iterating over the set you passed in. Defaults to Object. This type must be fully qualified.	C

CHAPTER

4 Navigation (Webflow) JSP Tags

The WebLogic Portal product suite includes a set of JSP tags designed to facilitate the development of JSPs using Webflow. Use of these predefined tags will eliminate the need for your JSPs to contain any Java code related to Webflow. This topic explains how to import this set of tags into your Web pages, and describes the purpose of each tag.

This topic includes the following sections:

- URL Creation Tags
- Form Tags
- Validated Form Tags
- Pipeline Session Tags

URL Creation Tags

The Webflow URL tags described in this section are used to create dynamic or static URLs for links and other resources within a JSP.

Note: In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<webflow:createWebflowURL>

The <webflow:createWebflowURL> tag (Table 4-1) is used in a JSP to dynamically create a Webflow URL in a JSP. The Webflow URL may include the protocol, domain name, port, Web application URI, WebflowServlet URI, and query string.

Tag Library	webflow.tld

Import Statement	`<%@ taglib uri="webflow.tld"`
	`   prefix="webflow" %>`

Class Implemented	CreateWebflowURLTag

Table 4-1 describes the <webflow:createWebflowURL> tag attributes.

Table 4-1 <webflow:createWebflowURL>

Tag Attribute	Required	Type	Description	R/C
domainName	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
doRedirect	No	String	Causes the WebflowServlet to perform a redirect instead of a forward to a presentation node. Valid values are true and false. The default is false (not to redirect).	R
encode	No	String	Informs Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs"for more information.	R

Table 4-1 <webflow:createWebflowURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 4-1 <webflow:createWebflowURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
`event`	Yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
`extraParams`	No	String	Used to supply additional request parameters as name/value pairs.	R

Table 4-1 <webflow:createWebflowURL> (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are calculate, http, and https.
			Calculate will yield HTTPS if any node in the origin/event branch chain list is matched under the HTTPS_URL_PATTERNS context parameter in the application's WEB-INF/web.xml file. Calculate is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for HTTPSIND_DEFAULT_VALUE in web.xml is used. If no value is found there, calculate is used.
			If an invalid value is used for this attribute, calculate is used.
			The following example shows a protocol set in web.xml. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VAL`
			`UE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of HTTP, HTTPS or CALCULATE can be specified in all uppercase or all lowercase.
				R

namespace	No	String	Indicates which Webflow configuration file the origin and event are defined in. If omitted, then the current (last successful) namespace is used.	R

Table 4-1 <webflow:createWebflowURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
`origin`	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
`pathPrefix`	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
`pathSuffix`	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R

Example

Listing 4-1 illustrates how to use the `<webflow:createWebflowURL>` JSP tag and its many attributes:

Note: It is recommended that you not hardcode values on a Web page.

Listing 4-1 Using `<webflow:createWebflowURL>` Tag and Attributes

```
<a href="<webflow:createWebflowURL event=’link.yo’ pathSuffix=’/morepath’
/>">A Path Suffix</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ pathPrefix=’/pathprefix’
/>">A Path Prefix</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ pathPrefix=’/pathprefix’
pathSuffix=’/suffix’ />">A Path Prefix and Path Suffix</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ domainName=’123.123.123.123’
/>">A Domain Name</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ pathSuffix=’/morepath’
extraParams=’sex=male’ />">A Path Suffix and One Extra Parameter</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ pathSuffix=’/morepath’
extraParams=’sex=male&animal=dog’ />">A Path Suffix and Two Extra Parameters</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ httpsInd=’https’ />">Always
Use HTTPS</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ httpsInd=’http’ />">Always Use
HTTP</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ httpsInd=’calculate’
/>">Calculate HTTPS</a>
<br>

<a href="<webflow:createWebflowURL event=’link.yo’ encode=’false’ />">Do Not
Encode the URL</a>
<br>
```

```
<a href="<webflow:createWebflowURL event='link.yo' doRedirect='true'
/>">Redirect, Instead of Forward</a>
<br>
```

<webflow:createResourceURL>

The <webflow:createResourceURL> tag (Table 4-2) is used in a JSP to create a static URL for a resource, using the value of the P13N_STATIC_ROOT context parameter in the application's WEB-INF/web.xml file. This tag may be used to load GIF images from a separate server.

Tag Library	webflow.tld
Import Statement	<%@ taglib uri="webflow.tld" prefix="webflow" %>
Class Implemented	CreateResourceURLTag

Table 4-2 describes the <webflow:createResourceURL> tag attributes.

Table 4-2 <webflow:createResourceURL>

Tag Attribute	Required	Type	Description	R/C
encode	No	Boolean	Informs Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_STATIC_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Table 4-2 <webflow:createResourceURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 4-2 <webflow:createResourceURL> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_STATIC_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
`resource`	No	String	Relative path to the file or image.	R

Example

Listing 4-2 illustrates how you can use the `<webflow:createResourceURL>` JSP tag to point to a specific resource, in this case, a `.gif` file.

Listing 4-2 Using `<webflow:createResourceURL>`

```
<img src="<webflow:createResourceURL
 resource="/images/button_checkout.gif"/>" border="0" alt="Proceed
 To Checkout" border="0"></a>
```

Form Tags

The Webflow JSP form tags described in this section are used to create simple dynamic links for form actions.

Note: In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<webflow:form>

The `<webflow:form>` tag is used in a JSP to dynamically generate an HTML form tag. This tag is not as sophisticated as the `<webflow:validatedForm>` tag, but is simpler.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri=”webflow.tld”`
	`prefix=”webflow” %>`

Class Implemented	WebflowFormTag

Note: This tag does not support the embedded Webflow form tags like the `<webflow:validatedForm>` does.

Table 4-3 lists the `<webflow:form>` tag attributes.

Table 4-3 <webflow:form>

Tag Attribute	Required	Type	Description	R/C
`domainName`	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R

Table 4-3 <webflow:form> (Continued)

Tag Attribute	Required	Type	Description	R/C
`doRedirect`	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R
`encode`	No	String	Informs Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See "Encoding Webflow URLs" for more information.	R

Table 4-3 <webflow:form> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 4-3 <webflow:form> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
`event`	Yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
`extraParams`	No	String	Used to supply additional request parameters as name/value pairs.	R
`hide`	No	String	If set to `false`, the namespace, origin, and event will be displayed on the command line instead of as hidden form fields. Valid values are `true` and `false`. The default value is `true`.	R

Table 4-3 <webflow:form> (Continued)

Tag Attribute	Required	Type	Description	R/C

`httpsInd`	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are `calculate`, `http`, and `https`.
			`Calculate` will yield HTTPS if any node in the origin/event branch chain list is matched under the `HTTPS_URL_PATTERNS` context parameter in the application's `WEB-INF/web.xml` file. `Calculate` is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for `HTTPSIND_DEFAULT_VALUE` in `web.xml` is used. If no value is found there, `calculate` is used.
			If an invalid value is used for this attribute, `calculate` is used.
			The following example shows a protocol set in `web.xml`. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALUE</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of `HTTP`, `HTTPS` or `CALCULATE` can be specified in all uppercase or all lowercase.
				R

`method`	No	String	The method to be used for the form. Valid values are `get` and `post`. The default value is `post`.	R
`name`	No	String	The name of the form.	R

Table 4-3 <webflow:form> (Continued)

Tag Attribute	Required	Type	Description	R/C
namespace	No	String	Indicates which Webflow configuration file the origin and event are defined in. If omitted, then the current (last successful) namespace is used.	R
origin	No	String	The node where the event will be coming from. Origins follow the node-name.node-type format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
pathPrefix	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
pathSuffix	No	String	Used to suffix the path information. The additional path information will be placed after the WebflowServlet URI. This information can then be retrieved via the request.getPathInfo() method.	R
htmlAttributes	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R

Example

Listing 4-3 illustrates how to use the `<webflow:form>` JSP tag:

Listing 4-3 Using <form>

```
<webflow:form event="button.go" >
   <input type="text" name="username" >
</webflow:form>
```

Validated Form Tags

The Webflow's validated form JSP tags are used to dynamically generate HTML forms that can be validated. These tags work in conjunction with an Input Processor and the ValidatedValues class, described in the *Javadoc*. When a Web site visitor enters invalid information, the visitor's input is preserved and redisplayed with an appropriate error message.

<webflow:validatedForm>

The <webflow:validatedForm> tag is used in a JSP to dynamically create the URL that defines the action in an HTML form. This tag should be used in conjunction with the com.bea.p13n.appflow.webflow.forms.* package and the other nested form tags defined in the webflow.tld file (which are described later in this section).

Tag Library	webflow.tld
Import Statement	<%@ taglib uri="webflow.tld" prefix="webflow" %>
Classes Implemented	WebflowValidatedFormTag WebflowValidatedFormTagExtraInfo

Table 4-4 describes the <webflow:validatedForm> tag attributes.

Table 4-4 <webflow:validatedForm>

Tag Attribute	Required	Type	Description	R/C

`applyStyle`	No	String	Applies the associated CCS style as indicated by the field status to the message, the field, or to none. Therefore, valid values are `message`, `field`, and `none`. The default value is `message`.
			See also:
			`invalidStyle`
			`styleId`
			`validStyle`
			`unspecifiedStyle`
				R

`domainName`	No	String	Used to change the domain name or IP address of the URL. This may be used if Webflow is fronted by a proxy server and that server resides on another machine.	R
`doRedirect`	No	String	Causes the `WebflowServlet` to perform a redirect instead of a forward to a presentation node. Valid values are `true` and `false`. The default is `false` (not to redirect).	R
`encode`	No	String	Informs Webflow to encode the URL. URLs need to be encoded if you wish to maintain session state and the browser does not accept cookies. Valid values are true and false. The default value is determined by the value of ENCODE_URLS set in the web app's web.xml file. URLs will only need to be encoded if the browser does not accept cookies. See “Encoding Webflow URLs” for more information.	R

Table 4-4 <webflow:validatedForm> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape	No	String	Tells Weblfow to escape the URL. URLs must be escaped if they will contain any characters (with some exceptions - see below) that would be encoded using java.net.URLEncoder.encode(). Note however that even when escaping is on the entire URL is not encoded rather the URL will be tokenized using the characters ':', '/', '?', '=', and '&' then the substrings between the tokens are encoded using java.net.URLEncoder.encode(). The tokenizing is necessary so that the URL will still be recognized by the Webflow engine. Valid values for this attribute are 'NO', 'YES' or 'CALCULATE'. Escaping is relatively costly and should be avoided if possible but is required if the URL might have any characters other then those ignored by java.net.URLEncoder.encode(), that is any characters other than 'a' through 'z', 'A' through 'Z, '0' through '9', '-', '_', '.', or '*' with the exception of the tokenizing characters mentioned above. If the content of the URL will not be determined until runtime and might contain "illegal" characters you should use either 'YES' or 'CALCULATE'. 'CALCULATE' should be used with care.	

Table 4-4 <webflow:validatedForm> (Continued)

Tag Attribute	Required	Type	Description	R/C
escape (con't)	No	String	(Continued from previous page) For sites that have a small number of URLs that will need escaping using 'CALCULATE' rather then 'YES' will result in a performance improvement. But, since 'CALCULATE' first checks the URL then encodes if needed, sites where most URLs need escaping will have poorer performance using 'CALCULATE' rather then 'YES'. If omitted the default value for escaping is determined by the value of ESCAPE_URLS in the web app's web.xml file. See "Encoding Webflow URLs" for more information.	
`event`	Yes	String	Webflow will use this in combination with the origin to resolve a destination in the supplied namespace XML file.	R
`extraParams`	No	String	Used to supply additional request parameters as name/value pairs.	R
`hide`	No	String	If set to `false`, the namespace, origin, and event will be displayed on the command line instead of as hidden form fields. Valid values are `true` and `false`. The default value is `true`.	R

Table 4-4 <webflow:validatedForm> (Continued)

Tag Attribute	Required	Type	Description	R/C

httpsInd	No	String	Informs Webflow to calculate the protocol or use HTTPS or HTTP. Valid values are calculate, http, and https.
			Calculate will yield HTTPS if any node in the origin/event branch chain list is matched under the HTTPS_URL_PATTERNS context parameter in the application's WEB-INF/web.xml file. Calculate is more dynamic and expensive, but if the protocol needs to be forced you can specify it here.
			If a value is not set for this attribute, the context-param value for HTTPSIND_DEFAULT_VALUE in web.xml is used. If no value is found there, calculate is used.
			If an invalid value is used for this attribute, calculate is used.
			The following example shows a protocol set in web.xml. This value would be used if no JSP tag attribute is set.
			`<context-param>`
			`<param-name>HTTPSIND_DEFAULT_VALU`
			`E</param-name>`
			`<param-value>HTTPS</param-value>`
			`</context-param>`
			The possible values of HTTP, HTTPS or CALCULATE can be specified in all uppercase or all lowercase.
				R

invalidStyle	No	String	The CSS style used to format the HTML field or the message when the field is invalid.	R

Table 4-4 <webflow:validatedForm> (Continued)

Tag Attribute	Required	Type	Description	R/C
`messageAlign`	No	String	Indicates whether to align the error message above the field, to the right of the field, or below the field. Therefore, value values are `top`, `right`, and `bottom`. The default value is `right`.	R
`method`	No	String	The method to be used for the form. Valid values are `get` and `post`. The default value is `post`.	R
`name`	No	String	The name of the form.	R
`namespace`	No	String	Indicates which Webflow configuration file the origin and event are defined in. If omitted, then the current (last successful) namespace is used.	R
`origin`	No	String	The node where the event will be coming from. Origins follow the `node-name.node-type` format. This may or may not be equal to the page name. If omitted, then the JSP page name is used.	R
`pathPrefix`	No	String	Used to prefix the path information. This string will be placed in front of the Web application URI. This can be used when the proxy server is located on the same machine. **Note:** The proxy must strip the path prefix before forwarding the request to Webflow.	R
`pathSuffix`	No	String	Used to suffix the path information. The additional path information will be placed after the `WebflowServlet` URI. This information can then be retrieved via the `request.getPathInfo()` method.	R
`styleId`	No	String	Scripting variable that will be set to one of invalidStyle, unSpecifiedStyle, or validStyle, depending on the field’s status: `valid`, `invalid`, `unspecified`. Can be used for finer control of formatting the HTML form.	R

Table 4-4 <webflow:validatedForm> (Continued)

Tag Attribute	Required	Type	Description	R/C
`unspecifiedStyle`	No	String	Used to specify the intial CSS style of the HTML field before validation occurs.	R
`validStyle`	No	String	The CSS style used to format the HTML field when it is valid.	R
htmlAttributes	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R

Example

Listing 4-4 illustrates how to use the <webflow:validatedForm> JSP tag to dynamically create the URL that defines the action in an HTML form..

Listing 4-4 Using <webflow:validatedForm>

```
<%
  moreAttributes = "ENCTYPE=\"multipart/form-data\"";
%>
<p> use enctype="multipart/form-data" and hide="false"...
<br>
<webflow:validatedForm name="uploadFeedback"
  htmlAttributes='<%= moreAttributes %>'
  hide="false"
  event="uploadFeedback.submit"
  namespace="test_forms" >
<webflow:text size="50" name="testText"value="startingDefaultValue" />
  <input type="submit" name="SubmitButton" value="SubmitButtonLabel" />

</webflow:validatedForm>
```

<webflow:text>

The `<webflow:text>` tag is used in a JSP to validate an HTML text field. This tag must be nested in the `<webflow:validatedForm>` tag.

Tag Library	`webflow.tld`
Import Statement	`<%@ taglib uri="webflow.tld" prefix="webflow" %>`
Classes Implemented	TexTag

Table 4-5 describes the `<webflow:text>` tag attributes.

Table 4-5 <webflow:text>

Tag Attribute	Required	Type	Description	R/C
`htmlAttributes`	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R
`maxlength`	No	String	The maximum length of the text field.	R
`name`	Yes	String	The name of the text field.	R
`retainValue`	No	String	Determines whether or not the text field should retain its input upon redisplay. Valid values are `true` and `false`.	R
`size`	No	String	The size of the text field.	R
`style`	No	String	The HTML CSS style associated with the text field.	R
`value`	No	String	The initial value of the text field.	R

<webflow:password>

The `<webflow:password>` tag is similar to the `<webflow:text>` tag except that field input is masked with asterisks (****). This tag must be nested in the `<webflow:validatedForm>` tag.

Tag Library	`webflow.tld`

Import Statement

```
<%@ taglib uri=”webflow.tld”
   prefix=”webflow” %>
```

Classes Implemented PasswordTag

Table 4-6 <webflow:password>

Tag Attribute	Required	Type	Description	R/C
`htmlAttributes`	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R
`maxlength`	No	String	The maximum length of the password field.	R
`name`	Yes	String	The name of the password field.	R
`retainValue`	No	String	Determines whether or not the password field should retain its input upon redisplay. Valid values are `true` and `false`. The default value is `false`.	R
`size`	No	String	The size of the password field.	R
`style`	No	String	The HTML CSS style associated with the password field.	R
`value`	No	String	The initial value of the password field.	R

<webflow:radio>

The `<webflow:radio>` tag is used in a JSP to represent an HTML radio button, but preserves the input. This tag must be nested in the `<webflow:validatedForm>` tag.

Tag Library `webflow.tld`

Import Statement

```
<%@ taglib uri=”webflow.tld”
   prefix=”webflow” %>
```

Classes Implemented RadioTag

Table 4-7 describes the `<webflow:radio>` tag attributes.

Table 4-7 <webflow:radio>

Tag Attribute	Required	Type	Description	R/C
`checked`	No	String	Determines whether or not the radio button is initially selected. Valid values are `true` and `false`. The default value is `false`.	R
`htmlAttributes`	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R
`name`	Yes	String	The name of the radio button field.	R
`value`	Yes	String	The initial value of the radio button field.	R

<webflow:checkbox>

The `<webflow:checkbox>` tag is used in a JSP to generate the required HTML for a check box. This tag will preserve input upon the `InputProcessor` throwing an `InvalidFormFieldException`. This tag must be nested in the `<webflow:validatedForm>` tag.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri=”webflow.tld”`
	`prefix=”webflow” %>`

Classes Implemented	CheckboxTag

Table 4-8 describes the `<webflow:checkbox>` tag attributes.

Table 4-8 <webflow:checkbox>

Tag Attribute	Required	Type	Description	R/C
`checked`	No	String	Determines whether or not the check box field is initially selected. Valid values are `true` and `false`. The default value is `false`.	R

Table 4-8 <webflow:checkbox> (Continued)

Tag Attribute	Required	Type	Description	R/C
`htmlAttributes`	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R
`name`	Yes	String	The name of the check box field.	R
`value`	Yes	String	The initial value of the check box field.	R

<webflow:textarea>

The `<webflow:textarea>` tag is used in a JSP to represent an HTML text area, but preserves the input. This tag must be nested in the `<webflow:validatedForm>` tag.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri="webflow.tld"`
	`prefix="webflow" %>`

Classes Implemented	TextareaTag

Table 4-9 describes the `<webflow:textarea>` tag attributes.

Table 4-9 <webflow:textarea>

Tag Attribute	Required	Type	Description	R/C
`cols`	No	String	The number of columns for the text area.	R
`name`	Yes	String	The name of the text area.	R
`retainValue`	No	String	Determines whether or not the text area should retain its input upon redisplay. Valid values are `true` and `false`.	R
`rows`	No	String	The number of rows for the text area.	R
`style`	No	String	The HTML CSS style associated with the text area.	R

Table 4-9 <webflow:textarea> (Continued)

Tag Attribute	Required	Type	Description	R/C
`value`	No	String	The initial value of the text area.	R
`wrap`	No	String	Determines whether or not the text area should wrap input. Valid values are `true` and `false`. The default value is `true`.	R

<webflow:select>

The `<webflow:select>` tag is used in a JSP to represent a list box, but preserves its input. This tag must be nested in the `<webflow:validatedForm>` tag.

Tag Library	`webflow.tld`
Import Statement	`<%@ taglib uri="webflow.tld" prefix="webflow" %>`
Classes Implemented	SelectTag

Table 4-10 describes the `<webflow:select>` tag atttibutes.

Table 4-10 <webflow:select>

Tag Attribute	Required	Type	Description	R/C
`htmlAttributes`	No	String	Additional HTML attributes. Any attribute not supported directly can be supplied here.	R
`multiple`	No	String	Determines whether or not the list box allows multiple selections. Valid values are `true` and `false`. The default value is `false`.	R
`name`	Yes	String	The name of the list box.	R
`size`	No	String	The size of the list box.	R
`style`	No	String	The HTML CSS style attribute.	R

<webflow:option>

The `<webflow:option>` tag (Table 4-11) is used in a JSP to represent an HTML option, but preserves the input. An option is one value in a list box. This tag must be nested in the `<webflow:select>` tag.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri="webflow.tld"`
	`prefix="webflow" %>`

Classes Implemented	OptionTag

Table 4-11 describes the `<webflow:option>` tag attributes.

Table 4-11 <webflow:option>

Tag Attribute	Required	Type	Description	R/C
`selected`	No	String	Determines whether or not the option is initially selected. Valid values are `true` and `false`. The default value is `false`.	R
`style`	No	String	The HTML style attribute.	R
`value`	Yes	String	The value the option represents.	R

Example

Listing 4-5 uses each of the validated form tags in an HTML form page that gathers some information about a Web site’s visitors.

Listing 4-5 Using `webflow.tld` Form Tags

```
<%@ taglib uri=”webflow.tld” prefix=”webflow” %>

<% String validStyle="background: white; color: black; font-family: Arial"; %>
<% String invalidStyle="background: white; color: red; font-style: italic"; %>

<%-- If there was an InvalidFormDataException thrown, display the message --%>

<font size="5" color="green"><webflow:getException/></font>
<br>

<webflow:validatedForm event="button.go" applyStyle="message"
 messageAlign="right" validStyle="<%=validStyle%>"
 invalidStyle="<%=invalidStyle%>" unspecifiedStyle="<%=validStyle%>" >
<p>

Username:
<webflow:text name="username" value="start" size="15" maxlength="30"
 htmlAttributes="onMouseOver=\\"self.status='User ID';return true\\"" />
<br>

Password:
<webflow:password name="password" size="15" retainValue="true" maxlength="30"
 htmlAttributes="onMouseOver=\\"self.status='Secret Password';return true\\"" />
<br>

Sex:
<webflow:radio name="sex" checked="true" value="M"/>Male
<webflow:radio name="sex" value="F" />Female
<br>

Favorite Pet(s):
<webflow:checkbox name="cat" value="cat" />Cat <br>
<webflow:checkbox name="dog" checked="true" value="dog" />Dog <br>
<webflow:checkbox name="bird" value="bird" />Bird
<p>

Comment:
<webflow:textarea name="comment" cols="20" rows="3" value="hello" />
<br>
```

```
Hobbies:
<webflow:select name=”hobbies" size="3" multiple="true">
   <webflow:option value="Running"/>Running
   <webflow:option value="Skiing"/>Skiing
   <webflow:option value="Motocross"/>MotoX
   <webflow:option value="Rugby"/>Rugby
</webflow:select>
<br>

<input type="submit" name="Submit"/>

</webflow:validatedForm>
```

Pipeline Session Tags

A Pipeline Session is used to share information between Input Processors, Pipeline Components, and presentation nodes. The Pipeline Session JSP tags are used to retrieve and set properties in the Pipeline Session. Presentation nodes (such as JSPs) are typically used to retrieve information from the Pipeline Session, while Input Processors and Pipeline Components place properties into the Pipeline Session. There are, however, JSP tags for setting properties in the Pipeline Session.

<webflow:setProperty>

The `<webflow:setProperty>` tag sets a property in the Pipeline Session.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri=”webflow.tld”`
	`prefix=”webflow” %>`

Classes Implemented	SetPropertyTag

Table 4-12 describes the <webflow:setProperty> tag attributes.

Table 4-12 <webflow:setProperty>

Tag Attribute	Required	Type	Description	R/C
namespace	No	String	Use the namespace attribute to force webflow to use a particular webflow configuration file defining a specific origin and event. If omitted then the current namespace (last successful namespace) is used.	R
property	Yes	String	Name or key with which the given property is to be associated.	R
scope	No	String	The scope of the property. Valid values are session and request. The default value is session.	R
value	Yes	Object	The value to associate with the property, specified as an object name or Java expression.	R

Example

Listing 4-6 illustrates how to use the <webflow:setProperty> JSP tag to set some arbitrary object in the Pipeline Session (Request-scoped):

Listing 4-6 Using <webflow:setProperty>

```
<% SomeObject so = new SomeObject("TWO"); %>
<webflow:setProperty property="myobject" value="<%= so %>"
scope="request" />
```

<webflow:getProperty>

The <webflow:getProperty> tag (Table 4-13) retrieves a named property from the Pipeline Session. This tag can be inlined or return a scripting variable.

Tag Library	webflow.tld
Import Statement	<%@ taglib uri="webflow.tld" prefix="webflow" %>
Classes Implemented	GetPropertyTag

Table 4-13 describes the <webflow:getProperty> tag attributes.

Table 4-13 <webflow:getProperty>

Tag Attribute	Required	Type	Description	R/C
namespace	No	String	Use the namespace attribute to force webflow to use a particular webflow configuration file defining a specific origin and event. If omitted then the current namespace (last successful namespace) is used.	R
id	No	Object	Java scripting variable to receive the instance of the returned object. If omitted, the toString() method will be called on the object and the results will be displayed in the browser.	R
property	Yes	String	The name or key of the property to obtain from the Pipeline Session.	R
scope	No	String	The scope of the property, which can be request or session. Request-scoped properties can improve performance, especially in a cluster because they do not need to be replicated. Valid values are session and request. The default value is session.	R

Table 4-13 <webflow:getProperty> (Continued)

Tag Attribute	Required	Type	Description	R/C
`type`	No	String	A Java class name, which can be used to cast your exception.	R

Example 1

The following code sample shows how you can use the `<webflow:getProperty>` JSP tag inline. The `toString()` method is called on the instance of `SomeObject`:

```
result = <webflow:getProperty property="myobject" scope="request"
/>
```

Example 2

Listing 4-7 on page 4-34 shows how to use the `<webflow:getProperty>` JSP tag to return a scripting variable of type `SomeObject`.

Listing 4-7 Using <webflow:getProperty>; Example 2

```
<webflow:getProperty id="myObj" property="myobject"
type="com.bea.test.SomeObject" scope="request" />
result = <%= myObj.getValue() %>
```

<webflow:setValidatedValue>

The `<webflow:setValidatedValue>` tag (Table 4-14) is used in a JSP to configure the display of fields in a form that a Web site visitor must correct. Usually this is done within an Input Processor, but it can also be done from a JSP by using this tag. The `<webflow:setValidatedValue>` tag is used in tandem with the `<webflow:getValidatedValue>` tag.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri="webflow.tld" prefix="webflow" %>`
Classes Implemented	setValidatedValueTag

Note: You might want to consider using the `<webflow:validatedForm>` tags instead of `<webflow:setValidatedValue>` as it supports the `validatedValues` class from previous releases. However, if some low-level functionality needs to be accessed, then these tags are still valid.

Table 4-14 describes the `<webflow:setValidatedValue>` tag attributes.

Table 4-14 <webflow:setValidatedValue>

Tag Attribute	Required	Type	Description	R/C
`fieldName`	No	String	The name of the field for which the status is desired. This should match the HTML form field name.	R
`fieldStatus`	No	String	The processing status of the field. Valid values are: `unspecified`—Field was left blank; Web site visitor must enter some data. `invalid`—Data is entered incorrectly. `valid`—Data is entered correctly.	R
`fieldValue`	No	String	The new value of the field.	R

Example

When used in a JSP, this sample code will obtain the current value and processing status of the `<field_name>` form field.

```
<webflow:setValidatedValue fieldName="<field_name>"
fieldValue="<field_value>" fieldStatus="status" />
```

<webflow:getValidatedValue>

The <webflow:getValidatedValue> tag) is used in a JSP to display the fields in a form that a Web site visitor must correct. The <webflow:getValidatedValue> tag is used in tandem with the<webflow:setValidatedValue> tag.

Tag Library	webflow.tld
Import Statement	<%@ taglib uri="webflow.tld" prefix="webflow" %>
Classes Implemented	getValidatedValueTag

Note: You may want to consider using the <webflow:validatedForm> tags instead of <webflow:getValidatedValue>, as it supports the ValidatedValues class from previous releases. However, if some low-level functionality needs to be accessed, then these tags are still valid.

Table 4-15 describes the <webflow:getValidatedValue> tag attributes.

Table 4-15 <webflow:getValidatedValue>

Tag Attribute	Required	Type	Description	R/C
fieldColor	No	String	Scripting variable set to one of invalidColor, validColor, or unspecifiedColor (depending on the status). This can be used to change the color of the field or message.	R
fieldDefaultValue	No	String	The default value to use if the fieldValue is missing.	R
fieldMessage	No	String	A scripting variable used to provide a specific message for the current field.	R
fieldName	Yes	String	The name of the field for which the status is desired. This should match the HTML form field name.	R

Table 4-15 <webflow:getValidatedValue> (Continued)

Tag Attribute	Required	Type	Description	R/C
`fieldStatus`	Yes	String	The status of the field. Valid values are: `unspecified`—Field was left blank; Web site visitor must enter some data. `invalid`—Data is entered incorrectly. `valid`—Data is entered correctly.	R
`fieldValue`	Yes	String	Scripting variable representing the value of the form field.	R
`invalidColor`	No	String	The color with which the label for an invalid field is to be marked. Defaults to `red`.	R
`unspecified Color`	No	String	If the Web site visitor leaves a required field blank, this will be the color of the label for that field. Defaults to `red`.	R
`validColor`	No	String	The color with which the label for a valid field is to be marked. Defaults to `black`.	R

These fields are determined and marked by an Input Processor after performing its validation activities. All `InputProcessors` use a `ValidatedValues` object to communicate which fields were successfully processed as well as those that were determined to be invalid. For more information, see the *Javadoc*.

Example 1

When used in a JSP, this sample code will obtain the current value and processing status of the `<field_name>` form field.

```
<webflow:getValidatedValue fieldName=”<field_name>”
fieldValue=”<field_value>” fieldStatus=”status” />
```

Example 2

The `<webflow:getValidatedValue>` tag refers to the `webflow.tld` tag library to retrieve available elements/attributes. In this example, a request is being made to obtain the following values from the HTTP session:

```
fieldName
```

```
fieldValue
fieldStatus
validColor
invalidColor
unspecifiedColor
fieldColor
```

These attributes are used for display purposes. (In this case, indicate that this field is required and mark it in red.) The overall goal is to display values back to the Web site visitor, indicating which pieces are valid/invalid as returned from the Input Processor, as shown in Listing 4-8.

Listing 4-8 Using `<webflow:getValidatedValue>`; Example 2

```
<webflow:getValidatedValue
fieldName="<%=HttpRequestConstants.CUSTOMER_FIRST_NAME%>"
fieldValue="customerFirstName" fieldStatus="status"
validColor="black"
invalidColor="red" unspecifiedColor="black" fieldColor="fontColor"
/>
```

<webflow:getException>

The `<webflow:getException>` tag is used to retrieve the exception or message thrown by a Webflow processor. This can be the message associated with an `InvalidFormFieldException` exception or a `ProcessingException` exception. This tag can be inlined (in which it calls the `getMessage()` method on the exception) or return a scripting variable representing the exception.

Tag Library	`webflow.tld`

Import Statement	`<%@ taglib uri="webflow.tld"`
	`prefix="webflow" %>`

Classes Implemented	getExceptionTag

Table 4-16 describes the `<webflow:getException>` tag attributes.

Table 4-16 <webflow:getException> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
`id`	Yes	Exception	Java scripting variable, which can be used to retrieve an instance of the exception.	R
`type`	No	String	Java class name, which can be used to cast the exception.	R

Example

Listing 4-9 illustrates how to use the `<webflow:getException>` JSP tag:

Listing 4-9 Using <webflow:getException>

```
<%-- If there was an InvalidFormDataException thrown, display the
message --%>
<font size=”5" color="red"><webflow:getException/></font>
```

CHAPTER

5 Catalog Development JSP Tags

The Commerce services included in the WebLogic Portal product suite provide JavaServer Page (JSP) templates and JSP tags that implement commonly used Web-based product catalog features. The Product Catalog JSP templates allow your customers to search for product items or browse through categories to locate items; the JSP tags are used to implement this functionality.

This topic includes the following sections:

- Catalog JSP Tags
- E-Business JSP Tags

Catalog JSP Tags

This section summarizes the tags that comprise the Product Catalog JSP tag library. These tags are used in the JSP templates that comprise the default Product Catalog. You can add or remove tags in your use of the JSP templates to match your specific formatting requirements.

Note: In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<catalog:getProperty>

Tag Library	`cat.tld`
Import Statement	`<%@ taglib uri="cat.tld" prefix="catalog" %>`
Class Implemented	GetPropertyTag GetPropertyTagExtraInfo

Use the `<catalog:getProperty>` tag (Table 5-1) to retrieve a property for display from either a `ProductItem` or `Category`. The property can either be an explicit property (a property that can be retrieved using a `get` method on the Catalog item) or an implicit property (a property available through the `ConfigurableEntity getProperty` methods on the Catalog item). The tag first checks to see if the specified property can be retrieved as an explicit property. If it cannot, the specified property is retrieved as an implicit property.

Table 5-1 describes the `<catalog:getProperty>` tag attributes.

Table 5-1 <catalog:getProperty> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
`getterArgument`	No	String	Denotes a reference to an object supplied as an argument to an explicit property getter method. May also be used to obtain implicit or custom properties that are defined using the property set framework, in which case the `getterArgument` would be the scope name for the property set (see second example below). The object must be presented in the form `<%= getterArgumentReference %>` and must be a run-time expression.	R

Table 5-1 <catalog:getProperty> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
`id`	No	String	`id="newInstance"` If the `id` attribute is supplied, the value of the retrieved property will be available in the variable name to which `id` is assigned. Otherwise, the value of the property is inlined.	C
`object`	Yes	Catalog item	Denotes a reference to a `ProductItem` or `Category` object that must be presented in the form `<%= objectReference %>.`	R
`propertyName`	Yes	String	`propertyName="propertyName"` Name of the property to retrieve. If the property is explicit, it may be one of the values shown in Table 5-2.	C
`returnType`	No	String	`returnType="returnType"` If the `id` attribute is supplied, declares the type of the variable specified by the `id` attribute.	C

Table 5-2 propertyName Values

Property Name	Catalog Item Type
`"contributor | coverage | creationDate | creator | description | image | key | language | modifiedDate | name | publisher | relation | rights | source"`	Catalog Item (common properties)
`"jsp"`	Category
`"availability | currentPrice | format | jsp | msrp | shippingCode | taxCode | type | visible"`	ProductItem

Example 1

Listing 5-1 retrieves the detail JSP information from an existing `ProductItem`:

Listing 5-1 Using `<catalog:getProperty>; Example 1`

```
<%@ taglib uri="cat.tld" prefix="catalog" %>

<catalog:getProperty
object="<%= item %>"
   propertyName="Jsp"
   getterArgument=
   "<%= new Integer(ProductItem.DETAILED_DISPLAY_JSP_INDEX) %>"
   id="detailJspInfo"
returnType="com.beasys.commerce.ebusiness.catalog.JspInfo"
/>
```

Example 2

The following example shows how to use the `getterArgument` attribute to obtain an implicit or custom property for a property set/schema with the following characteristics:

- Name: `MyCatalog`
- PropertyName: `color`

Note: Because the `getterArgument` must be a run-time expression, we assign `MyCatalog` to a String variable and use the variable as the value to the `getterArgument`.

Listing 5-2 Using `<catalog:getProperty>`; Example 2

```
<%@ taglib uri="cat.tld" prefix="catalog" %>

<%
String myPropertySetName = "MyCatalog";
ProductItem myProductItem = .....; // reference to a ProductItem
%>
<catalog:getProperty
   object="<%=myProductItem%>
   propertyName="color"
   getterArgument="<%=myPropertySetName%>"
/>
```

<catalog:iterateViewIterator>

Tag Library	`cat.tld`
Import Statement	`<%@ taglib uri="cat.tld" prefix="catalog" %>`
Class Implemented	IterateViewIteratorTag IterateViewIteratorTagExtraInfo

Use the `<catalog:iterateViewIterator>` tag to iterate through a `ViewIterator`. A `ViewIterator` is an iterator over a potentially large collection of remote data that is broken up into a series of fixed sized `Views`. `ViewIterators` are returned from all Catalog service API methods that may potentially return a large set of `ProductItems` or `Categories`. This tag allows you to iterate the `ViewIterator` one item (`ProductItem` or `Category`) at a time (the default behavior) or by an entire `View` (fixed size set of `ProductItems` or `Categories`) at a time. It is important to note that this tag does not reset the state of the `ViewIterator` upon completion.

Table 5-3 describes the `<catalog:iterateViewIterator>` tag attributes.

Table 5-3 <catalog:iterateViewIterator> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
`id`	Yes	String	`id="newInstance"` The value of the current iterated object will be available in the variable name to which the `id` is assigned.	C
`iterator`	Yes	ViewIterator	Denotes a reference to a `ViewIterator` object. Must be presented in the form `<%= iteratorReference %>`.	R

Table 5-3 <catalog:iterateViewIterator> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
iterateByView	No	String	iterateByView=”{true|false}” Specifies whether to iterate the ViewIterator by View or by Catalog item. If not specified, the ViewIterator will be iterated by Catalog item.	C
returnType	No	String	returnType=”returnType” Declares the type of the variable specified by the id attribute. Defaults to java.lang.Object. If iterateByView is true, the type is assumed to be com.beasys.commerce.ebusiness.catalog.View.	C

Example 1

Listing 5-3 displays the keys of all Categories in a ViewIterator:

Listing 5-3 Using <catalog:iterateViewIterator>; Example 1

```
<%@ taglib uri=”cat.tld” prefix=”catalog” %>

<catalog:iterateViewIterator
   iterator="<%= myIterator %>"
   id="category"
   returnType="com.beasys.commerce.ebusiness.catalog.Category">
   <%= category.getKey().toString() %>
</catalog:iterateViewIterator>
```

Example 2

The following example displays all the Views contained within a ViewIterator:

Listing 5-4 Using `<catalog:iterateViewIterator>`; Example 2

```
<%@ taglib uri="cat.tld" prefix="catalog" %>

<catalog:iterateViewIterator
   iterator="<%= myIterator %>"
   id="view"
   returnType="com.beasys.commerce.ebusiness.catalog.ViewIterator"
   iterateByView="true">
   <%= view.toString() %>
</catalog:iterateViewIterator>
```

<catalog:iterateThroughView>

Tag Library	`cat.tld`
Import Statement	`<%@ taglib uri="cat.tld" prefix="catalog" %>`
Class Implemented	IterateThroughViewTag IterateThroughViewTagExtraInfo

The `<catalog:iterateThroughView>` tag (Table 5-4) iterates through a `View` of a specified `ViewIterator`. The tag will iterate the `View` one Catalog item at a time until the end of the `View` is reached. If you do not specify a specific `View` (by index) through which to iterate, the current `View` of the `ViewIterator` is used. It is important to note that this tag does not reset the state of the `ViewIterator` upon completion.

Table 5-4 describes the `<catalog:iterateThroughView>` tag attributes.

Table 5-4 <catalog:iterateThroughView> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
`id`	Yes	String	`id="newInstance"` The value of the current iterated object will be available in the variable name to which the `id` is assigned.	C

Table 5-4 <catalog:iterateThroughView> Tag Attributes

Tag Attribute	Required	Type	Description	R/C
`iterator`	Yes	ViewIterator	Denotes a reference to a `ViewIterator` object that must be presented in the form `<%= iteratorReference %>`	R
`returnType`	No	String	`returnType="returnType"` Declares the type of the variable specified by the `id` attribute. Defaults to `java.lang.Object`.	C
`viewIndex`	No	Integer	Specifies the index of the View (relative to the start of the `ViewIterator`) through which to iterate. The referenced object must be presented in the form `<%= viewIndexIntegerReference %>`.	R

Example 1

Listing 5-5 displays the keys of all the `ProductItems` contained in the current `View` of a specified `ViewIterator`:

Listing 5-5 Using `<catalog:iterateThroughView>`; Example 1

```
<%@ taglib uri="cat.tld" prefix="catalog" %>

<catalog:iterateThroughView
   iterator="<%= myIterator %>"
   id="item"
   returnType="com.beasys.commerce.ebusiness.catalog.ProductItem">
<%= item.getKey().toString() %>
</catalog:iterateThroughView>
```

Example 2

The following example displays the keys of all the `ProductItems` contained in the first `View` of a specified `ViewIterator`:

Listing 5-6 Using `<catalog:iterateThroughView>`; Example 2

```
<%@ taglib uri=”cat.tld” prefix=”catalog” %>

<catalog:iterateThroughView
   iterator="<%= myIterator %>"
   id="item"
   returnType="com.beasys.commerce.ebusiness.catalog.ProductItem"
   viewIndex=”new Integer(0)”>
   <%= item.getKey().toString() %>
</catalog:iterateThroughView>
```

E-Business JSP Tags

This section summarizes the tags that comprise the E-Business JSP tag library.

<eb:smnav>

Tag Library	`eb.tld`
Import Statement	`<%@ taglib uri=”eb.tld” prefix=”eb” %>`
Class Implemented	ScrollableModelTag

A Scrollable Model is used to retrieve value objects so that only what is viewed is retrieved. The `<eb:smnav>` tag (Table 5-5) allows you to control the presentation of elements in the list of value objects that are being viewed, and provides links to the previous and next pages.

The `<eb:` preface stands for e-business. The Scrollable Model can be use throughout the e-business package to iterate through a list of objects. It can be used in conjunction with transaction, shopping cart, order history, or shipping services.

This tag relies on a Pipeline Session containing a ScrollableModel object on the `PipelineSessionConstants.SCROLLABLE_MODEL` key.

Table 5-5 describes the `<eb:smnav>` tag attributes.

Table 5-5 <eb:smnav> Tag Atttibutes

Tag Attribute	Required	Type	Description	R/C
`event`	No	String	The name of the link configurable in the Webflow as the visitor clicks on Next or Previous.	C
`nextstring`	No	String	The localized name for Next. Could be as simple as “>”.	C
`origin`	No	String	The current JSP page.	C
`pageindex`	No	String	The index of the page to display.	R
`prevstring`	No	String	The localized name for Previous. Could be as simple as “<”.	C

Example

The `orderhistory.jsp` that is part of the Commerce services JSP templates allows a visitor to browse page by page over the set of orders placed. Only 10 orders are displayed at a time. To go to the next or to the previous page, the visitor clicks on the “Next” or “Previous” hyperlinks shown by the tag. In Listing 5-7, if the visitor has 40 orders and is viewing the second page, the tag will be displayed as “Previous | 20-29 | Next”.

Listing 5-7 Using `<eb:smnav>`

```
<%@ taglib uri="eb.tld" prefix="eb" %>

<!-- Show the Previous / 10-19 / Next navigation string -->

<eb:smnav origin="orderhistory.jsp" event="link.viewOrderHistory"
 prevstring="Previous" nextstring="Next"
 pageindex="<%=pageIndexString%>" />
```

CHAPTER

6 Event and Behavior Tracking JSP Tags

This tag library contains several tag extensions used in BEA WebLogic Portal. Tags in this library are specifically used in the Events and Behavior Tracking service.

You can use events tags with scenario actions in promotions and campaigns. When Behavior Tracking is activated, you can track user behavior as users navigate across your site. Behavior Tracking records events to relational databases, which may be analyzed by third-party analytical tools.

The Events tags are divided into two general areas: content tracking and product tracking. Content and product tracking tags can be used in any Web or portal application.

This topic includes the following sections:

- Content Tags
- Product Tags

Content Tags

Note: In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<tr:clickContentEvent>

Tag Library	`tracking.tld`
Import Statement	`<%@ taglib uri="tracking.tld" prefix="tr" %>`
Class Implemented	ClickContentTag ClickContentExtraInfo

The `<tr:clickContentEvent>` tag is used to generate a behavior event when a user has clicked (through) on an ad impression. This tag will return a URL query string containing event parameters. It is then used when forming the complete URL that hyperlinks the content.

Table 6-1 lists the `<tr:clickContentEvent>` tag attributes.

Table 6-1 `<tr:clickContentEvent>`Attributes

Tag Attribute	Req'd	Type	Description	R/C
`documentId`	No	String	ID of the item that is displayed, if applicable (that is, an image URL or banner ad ID).	R
`documentType`	No	String	Type or category of the item that is displayed (if applicable).	R
`id`	No	String	Page variable which will hold the output of this tag.	C
`userId`	No	String	Name of the user that content was retrieved for. If the optional value is not provided, it will be set to the value of the `request.getRemoteUser()`.	R

Example

The example below demonstrates a clickthrough example going to the Webflow servlet. This link causes a clickthrough content event to be generated and also display the indicated content. The example shows how to generate a click content event after the user clicks a product description link. The default Webflow servlet's `<filter>`

tag, specified in the application's `web.xml` file, generates a call to the `ClickThroughEventFilter.doFilter()` method. This method checks for `ClickThroughConstants.EVENT_TYPE` in the `HttpRequest`, and then fires the click event if it is present.

The `ClickThroughConstants.EVENT_TYPE` is generated by adding the `<tr:clickContentEvent>` tag in the JSP, as shown in Listing 6-1.

Listing 6-1 Using `<tr:clickContentEvent>`

```
<tr:clickContentEvent documentId="<%= documentId %>"
  documentType="<%= documentType %>"
  userId="<%= userId %>"
  id="outputFromTag"
/>
```

The following associates the desired content with a link that references the output from the above tag.

Listing 6-2 Associating Content With a Link That References Output From Listing 6-1

```
<A HREF="<webflow:createWebflowURL event="link.clickContent"
   namespace="trackingWebApp_main" extraParams="<%= outputFromTag
   %>" />">Click Here to generate the clickContentEvent.</A>
```

Note: To redirect the user to another site, use `redirect="true"` in the `createWebflowURL` tag.

<tr:displayContentEvent>

The `<tr:displayContentEvent>` tag generates a behavior event when a user has received (viewed) an ad impression. “Ads” can be any HTML content, such as images, text, PDF files, and Web-compatible multimedia content.

Tag Library	`tracking.tld`
Import Statement	`<%@ taglib uri="tracking.tld" prefix="tr" %>`
Class Implemented	DisplayContentEventTag

Table 6-2 lists the `<tr:displayContentEvent>` tag attributes.

Table 6-2 <tr:displayContentEvent>

Tag Attribute	Req’d	Type	Description	R/C
`documentId`	No	String	ID of the item that is displayed, if applicable (that is, an image URL or banner ad ID).	R
`documentType`	No	String	Type or category of the item that is displayed (if applicable).	R

Example

The example below shows a code snippet of processing that would follow a `<cm:select>` call. For each document returned but not displayed in this example, the `<tr:displayContentEvent>` tag generates an event and passes the document's ID and type.

Listing 6-3 Using <tr:displayContentEvent>

```
<%@ taglib uri="tracking.tld" prefix="tr" %>
.
.
.
<es:forEachInArray id="nextRow" array="<%=headlines%>"
  type="com.bea.p13n.content.Content">
  <es:notNull item="<%=nextRow%>">
    <tr:displayContentEvent
      documentId="<%=nextRow.getIdentifier()%>"
      documentType="<%=headingProp%>"/>
  </es:notNull>
</es:forEachInArray>
```

Product Tags

Note: In the following tables, the Required column specifies if the attribute is required (yes) or optional (no). In the R/C column, C means that the attribute is a Compile time expression, and R means that the attribute can be either a Request time expression or a Compile time expression.

<trp:clickProductEvent>

Tag Library	`productTraking.tld`

Import Statement	`<%@ taglib uri="productTracking.tld"`
	`prefix="trp" %>`

Class Implemented	ClickProductEventTag

The `<trp:clickProductEvent>` tag is used to generate a behavior event when a user has clicked (through) on a product impression. This tag will return a URL query string containing event parameters. It is then used when forming the complete URL that hyperlinks the content.

At least one of `sku`, `categoryId`, or `documentId` is required.

Table 6-3 describes the `<trp:clickProductEvent>` tag attributes.

Table 6-3 <trp:clickProductEvent>

Tag Attribute	Req'd	Type	Description	R/C
`applicationName`	No	String	The webApp or application name, if applicable. Can be used to separate data when multiple storefronts are hosted on the same server (or persisted to the same database).	R
`categoryId`	No	String or Category object	Category of the product associated with the content displayed, if applicable.	R
`documentId`	Yes	String	Name of the item that is displayed, if applicable (that is, an image URL or banner ad ID).	R
`documentType`	No	String	Type or category of the item that is displayed (if applicable).	R
`sku`	No; see description	String or ProductItem object	ID of the product associated with the content item that is displayed, if applicable. `sku` is not normally required unless neither categoryId nor documentId is specified.	R
`userId`	No	String	Name of the user that content was retrieved for. If the optional value is not provided, it will be set to the value of the `request.getRemoteUser()`.	R

Example

The example below demonstrates a clickthrough example going to the Webflow servlet. This link will cause a clickthrough content event to be generated and also display the indicated content. This example shows how to generate a `ClickProductEvent` having a document ID using the product name (`productItem.getName()`) and SKU of the product's identifier.

Listing 6-4 Using <trp:clickProductEvent>

```
<%@ taglib uri="productTracking.tld" prefix="trp" %>
.
.
.
<%
detailsUrl = WebflowJSPHelper.createWebflowURL(pageContext,
"itemsummary.jsp", "link(" + detailsLink + ")",
"&" + HttpRequestConstants.CATALOG_ITEM_SKU + "=" +
productItem.getKey().getIdentifier() + "&" +
HttpRequestConstants.CATALOG_CATEGORY_ID + "=" +
category.getKey().getIdentifier() + "&" +
HttpRequestConstants.DOCUMENT_TYPE + "=" + detailsLink, true);
%>
<trp:clickProductEvent
   id="url"
   documentId="<%= productItem.getName() %>"
   sku="<%= productItem.getKey().getIdentifier() %>" />
<%
detailsUrl = detailsUrl + "&" + url;
%>
<a href="<%= detailsUrl %>">
```

<trp:displayProductEvent>

Tag Library	`productTraking.tld`

Import Statement	`<%@ taglib uri="productTracking.tld"`
	`prefix="trp" %>`

Class Implemented	DisplayProductEventTag

The <trp:displayProductEvent> tag is used to generate a behavior event when a user has received (viewed) a product impression, (typically a gif image).

At least one of sku, categoryId, or documentId is required.

Table 6-4 describes the <trp:displayProductEvent> tag attributes.

Table 6-4 `<trp:displayProductEvent>` Tag Attributes

Tag Attribute	Req'd	Type	Description	R/C
applicationName	No	String	The webApp or application name, if applicable. Can be used to separate data when multiple storefronts are hosted on the same server (or persisted to the same database).	R
categoryId	No	String or Category object	Category of the product associated with the content displayed, if applicable.	R
documentId	No	String	Name of the item that is displayed, if applicable (that is, an image URL or banner ad ID).	R
documentType	No	String	Type or category of the item that is displayed (if applicable). Suggestions: DisplayProductEvent.CATEGORY_BROWSE DisplayProductEvent.ITEM_BROWSE DisplayProductEvent.CATEGORY_VIEW DisplayProductEvent.BANNER_AD_PROMOTION	R
sku	No; see description	String or ProductItem object	ID of the product associated with the content item that is displayed, if applicable. sku is not normally required unless neither categoryId nor documentId is specified.	R

Example

Listing 6-5 shows an example of code that would follow the retrieval of a catalog item. The <tr:displayProductEvent> tag generates an event and passes the document's ID, type and SKU number of the product item.

Listing 6-5 Using `<tr:displayProductEvent>`

```
<%@ taglib uri="productTracking.tld" prefix="trp" %>
...
<trp:displayProductEvent
    documentId="<%= item.getName() %>"
    documentType="<%= DisplayProductEvent.ITEM_BROWSE %>"
    sku="<%= item.getKey().getIdentifier() %>" />
```

A JSP Templates

WebLogic Portal provides a set of perdefined JSP templates that you can use in your applications. You can either use the templates exactly as BEA provides them or you can modify them to meet your specific needs; for example by applying your own skins and layouts. This section lists the templates available.

This section includes information on the following subjects:

- Template Location
- Commerce Templates
- Campaign Templates
- Catalog Templates
- Catalog and Campaign Includes
- Order Templates
- Registration Templates
- User Templates

Template Location

All templates are located in the following directory:

```
<BEA_HOME>\weblogic700\portal\samples\portal\wlcsDomain\
   beaApps\wlcsApp\wlcs\commerce
```

Commerce Templates

Table A-1 lists the templates used for developing commerce services.

Table A-1 /commerce Templates

main.jsp	Anonymous main home - start page.
login.jsp	Provides form-based submission of username and password to gain access to account, and a link to create a new user.
badlogin.jsp	Similar to login.jsp, displays an error message (includes login.jsp).
configurationerror.jsp	Used when there is an error in the webflow configuration or pipeline.
newuserforward.jsp	Redirects to the user/newusercreation.jsp.
servererror.jsp	All-purpose 403, 404, 405, 500 server error page.
sessiontimeout.jsp	Session timeout page.
shoppingcart.jsp	Presents the items ordered; user may change quantities, delete, continue.

Campaign Templates

Table A-2 lists the JSP templates available for setting up Campaign services.

Table A-2 /commerce/campaignTemplates

contractor_journal.jsp	Sample offsite page associated with a campaign example.
fixup_channel.jsp	Sample offsite page associated with a campaign example.

Catalog Templates

Table A-3 Describes the JSP templates available for setting up a Catalog service

Table A-3 /CatalogTemplates

browse.jsp	Successive drill-downs via browse of the catalog. Same template applied for subcategories and sub-sub-categories.
details.jsp	Displays all the selected metadata for a single item; includes backtrack nav to parent category.
search.jsp	Power search page; includes power search results.
searchresults.jsp	Results from a quick keyword search conducted anywhere in the site.

Catalog and Campaign Includes

Table A-4 describes the JSP `include` templates available for setting up a Catalog or Campaign service.

Table A-4 Catalog and Campaign `include` Templates

category.jsp	Creates the list of subcategories, and presents the item summary of any items at the parent (current) category level. If there are more than x-number, the list is continued with "next" and "previous" pages.
itemdetails.jsp	An include module that contains the item details presentation.
itemsummary.jsp	An include module that allows the repeat of multiple items on a search or browse page.

Table A-4 Catalog and Campaign `include` Templates (Continued)

navigation.jsp	An include module providing "back-track" ("breadcrumbs") navigation in the browse interface.
navigation2.jsp	An include module providing "back-track" ("breadcrumbs") navigation in the browse interface.
admin.inc	Used on all pages and presents the top red-and-black banner with links to the main Administration screen, to this template index, and to a *.jsp.html file for the current template.
header.inc	Used on all pages and presents a Home link. Depending on the page that included header.inc, also presents links for Search, View Cart, Log In (for users who have not logged in yet), or Log Out (for authenticated users).
footer.inc	Used on all pages.
leftside.inc	Presents quick look-up and a promotional ad; for authenticated users, also presents a personalized message to the user, customer profile link, order history link, and payment history link.
stylesheet.inc	Used in all pages, this is a cascading stylesheet that defines global paragraph and text styles for the site.
states.inc	Used in all forms where US states are listed as options in an input pop-up.
countries.inc	Used in all forms where countries are listed as options in an input pop-up.
editaddresstemplate.inc	Used in profile and order forms where the user can edit an existing shipping address.
editcctemplate.inc	Used in profile and order forms where the user can edit an existing credit card.
newaddresstemplate.inc	Used in profile and order forms where the user can supply a new shipping destination address.
newcctemplate.inc	Used in profile and order forms where the user can supply a new credit card for payment.

Table A-4 Catalog and Campaign `include` Templates (Continued)

newdemographictemplate.inc	Included into register/newuser.jsp and user/editdemographics.jsp; presents a form that allows user to enter or update demographic information. The demographic details provided can then be used by campaign-related features.

Order Templates

Table A-5 decribes the templates available for setting up an Order service.

Table A-5 /Order Templates

selectaddress.jsp	Existing destination address(es) presented; user may choose, edit, add new.
addaddress.jsp	If user does not find the desired shipping address in selectaddress.jsp - includes newaddresstemplate.jsp.
shipping.jsp	Choices for express, airmail, ups, etc., presented; user may also add special shipping instructions.
selecttaxaddress.jsp	Conditional upon certain jurisdictions; user may select a county, etc.; then continue with checkout.
payment.jsp	Choices for which (credit card) account to use for this transaction; user may choose, edit, add new.
paymentnewcc.jsp	User may supply new credit account info - includes newcctemplate.inc.
paymenteditcc.jsp	User may change the credit account info as selected in payment.jsp - includes editcctemplate.inc.
checkout.jsp	Entire transaction with items, address, shipping choice, account, tax and totals presented for final charge authorization.
confirmorder.jsp	After successful checkout, summary printed to confirm receipt of order.

Table A-5 /Order Templates

orderhistory.jsp	User may view a summary of past or pending orders; may select one to view its contents, shipping dates, and other details.
paymenthistory.jsp	User may view a summary of past orders organized by payment.
orderstatus.jsp	Individual orders are displayed with their items, costs, status, shipping address and other details.
systemerror.jsp	Displays an error due to service system failure in the transaction components.

Registration Templates

Table A-6 describes the JSP Templates available for setting up a Registration service.

Table A-6 Register Templates

newuser.jsp	Empty profile fields for all personal information. Also includes a demographic survey to record user profile information. If you are using the full WebLogic Portal license (which includes campaign services) this data can be used by campaigns.

User Templates

Table A-7 describes the JSP templates available for setting up a user.

Table A-7 User Templates

viewprofile.jsp	Presents the known personal information for the logged-in user, with buttons for modifying and adding information.

Table A-7 User Templates (Continued)

editdemographics.jsp	Allows registered user to change demographic information that was entered previously.
editprofile.jsp	Change personal info (name, billing address, phones, email).
changepassword.jsp	User may change their password.
profilenewcc.jsp	User may supply new credit account info - includes newcctemplate.inc.
profileeditcc.jsp	User may change the credit account info as selected in payment.jsp - includes editcctemplate.inc.
profilenewaddress.jsp	User may supply an additional new shipping address - includes newaddresstemplate.inc.
profileeditaddress.jsp	User may modify an existing shipping address in profile - includes editaddresstemplate.inc.
newusercreation.jsp	Displayed after the user registers successfully - provides links to view cart, check out, and return home.
secureMain.jsp	A secure page that simply forwards the successfully logged-in user back to main.jsp.
usercreationforward.jsp	Used to redirect the newusercreation.jsp page after creating the user. This is done because the request does not contain the user information unless you do a redirect via the response object to the proper URL. This usercreationforward.jsp creates a new request that has the authenticated user's information, which allows the campaigns to start.

B Tuning JSP Performance

When a customer requests a page on your e-business Web site, WebLogic Portal compiles the corresponding JSP into a servlet. In addition to servlets that come from compiled JSPs, WebLogic Portal provides a set of servlets for exchanging information between various components of the system.

This section provides information on tuning JSP compiling and updating. It includes information in the following subjects:

- Step 1: Precompile JSPs
- Step 2: Specify a Java Compiler for a Web Application
- Step 3: Adjust the Intervals for Checking JSP and Servlet Modifications

Step 1: Precompile JSPs

For each of your Web applications that you deploy, you can determine when WebLogic Portal compiles JSPs:

- Manually precompile JSPs using weblogic.jspc.
- You can specify that the application server precompiles all JSPs when you start the server. When you activate the precompile option, the server startup process checks for new or modified JSPs in the Web application and compiles them.

- You can specify that the application server compiles JSPs only when they are requested. With this option deactivated, the server starts quickly but must compile each new or modified JSP when you access it, causing a significant delay the first time you request a new or modified JSP.

The sample applications for WebLogic Portal samples are precompiled on installation.

Use the following steps. For more information, see the WebLogic Server XML elements documentation at http://e-docs.bea.com/wls/docs70/webapp/weblogic_xml.html, and the *WebLogic Server Administration Guide* at http://e-docs.bea.com/wls/docs70/adminguide/overview.html#1036999.

1. If your Web application is deployed as a WAR file, unjar it.
2. In the WebLogic Server console, navigate to the web application, click the Configuration tab, and mark Precompile. (The parameter name in the weblogic.xml file is `weblogic.jsp.precompile`.)
3. Save the file.
4. If appropriate, rejar the WAR file.
5. Restart the server.

Step 2: Specify a Java Compiler for a Web Application

The WebLogic Server Administration Console specifies a Java compiler for each server configuration. All applications that you deploy on a server use this compiler unless a Web application's `weblogic.xml` file specifies a different compiler.

To review the current Java compiler for your server, in the left pane of the WebLogic Server Administration Console, click a server. In the right pane, on the Configurations tab, click the Compilers subtab. To enter a new name, enter the pathname of the Java compiler that you want to use for the Web application. To deploy any modifications, restart the server.

For more information, see the WebLogic Server XML elements documentation at http://e-docs.bea.com/wls/docs70/webapp/weblogic_xml.html, and the *WebLogic Server Administration Guide* at http://e-docs.bea.com/wls/docs70/adminguide/overview.html#1036999.

Step 3: Adjust the Intervals for Checking JSP and Servlet Modifications

You can specify how frequently a server checks for modifications to JSPs and source files for other servlets in a Web application.

The sample Web applications check for modified JSPs each time a Web browser requests a JSP. Likewise, each time the server sends a request to a servlet in a sample Web application, it checks for any modifications to the servlet class files.

For your production Web site, you can decrease the amount of time in which WebLogic Portal serves JSPs and processes requests to servlets by increasing the intervals at which the server checks for modifications.

Although the server performs faster with higher values for the modification-check intervals, the higher values reduce sensitivity to changes in your source files. For example, you can set the server to check for JSP modifications every 10 minutes. After you change a JSP, it will take up to 10 minutes for the server to see the modifications.

This section includes information on the following subjects:

- Understand the Page-Check Intervals Properties
- Adjust the Intervals

Understand the Page-Check Intervals Properties

The `pageCheckSeconds` attribute determines the interval at which a server checks to see if JSP files in a Web application have changed and need recompiling. Each Web application defines this property separately in its `WEB-INF\weblogic.xml` file:

```
<jsp-param>
     <param-name>pageCheckSeconds</param-name>
     <param-value>1</param-value>
</jsp-param>
```

The page-check interval does not determine the frequency with which a server checks for updated content that is stored in the database and in a content management system. Instead, the TTL (time-to-live) settings for various caches determine the refresh rate for content. For example, if you set the page-check intervals to once a second, and you set the TTL for the content cache to 10 minutes, it can take up to 10 minutes for the server to see the new content, even though it is checking for new JSP source code every second. For information on setting TTL properties, see “Performance Tuning” in the *Development Guide* at http://edocs.bea.com/wlp/docs70/dev/x10ding.htm#1040480

Adjust the Intervals

To determine the optimal page-check and reload-servlet intervals for your production Web site do the following:

1. Establish performance baselines by testing WebLogic Portal performance with the interval set to `-1` (which specifies that the server never checks for modifications).

2. Test the performance with the interval set to various numbers of seconds. For example, set the interval to `600` seconds (10 minutes) and test the performance. Then set the interval to `900` seconds and test the performance.

3. Choose an interval that provides the best performance while checking for modifications to JSP files and servlet classes at a satisfactory rate.

C JSP Tag Reference (by Name)

This section lists all the JSP tags available for Portal development, sorted by tag name. To see a more detailed description of the tag, including tag attribute descriptions and code samples showing how to use the tag, click the tag name in Table C-1.

Table C-1 Portal Development JSP Tags by Name

Tag Name	Description	Tag Library
`<ad:adTarget>`	Send an ad query to the content management system. Unlike the <ph:placeholder> tag, the query in the `<ad:adTarget>` tag does not compete with other queries in an ad placeholder.	`ad.tld`
`<catalog:getProperty>`	Retrieves a property for display from either a `ProductItem` or `Category`. The property can either be an explicit property (a property that can be retrieved using a `get` method on the Catalog item) or an implicit property (a property available through the `ConfigurableEntity` `getProperty` methods on the Catalog item).	`cat.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<catalog:iterateThroughView>`	Iterates through a `View` of a specified `ViewIterator`. The tag will iterate the `View` one Catalog item at a time until the end of the `View` is reached.	`cat.tld`
`<catalog:iterateViewIterator>`	Iterate through a `ViewIterator`. A `ViewIterator` is an iterator over a potentially large collection of remote data that is broken up into a series of fixed sized `Views`.	`cat.tld`
`<cm:getProperty>`	Retrieves the value of the specified content metadata property into a variable specified by `resultId`. It does not have a body. If `resultId` is not specified, the value will be inlined into the page, similar to the `<cm:printProperty>` tag. .	`cm.tld`
`<cm:printDoc>`	Inlines the raw bytes of a `Document` object into the JSP output stream. This tag does not support a body and only supports `Document` objects. It does not differentiate between text and binary data.	`cm.tld`
`<cm:printProperty>`	Inlines the value of the specified content metadata property as a string. It does not have a body. This tag operates on any `ConfigurableEntity`, not just the `Content` object. However, it does not support `ConfigurableEntity` successors.	`cm.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
<cm:select>	Uses only the search expression query syntax to select content. It does not support or use a body. After this tag has returned the <es:forEachInArray> tag zero can be used to iterate over the array of Content objects. This tag supports generic Content via a ContentManager interface.	cm.tld
<cm:selectById>	Retrieves content using the Content's unique identifier. This tag does not have a body. This tag is basically a wrapper around the select tag. It works against any Content object which has a string-capable primary key.	cm.tld
<eb:smnav>	Retrieve value objects so that only what is viewed is retrieved. This tag allows you to control the presentation of elements in the list of value objects that are being viewed, and provides links to the previous and next pages	eb.tld
<es:convertSpecialChars>	Converts characters which would normally signify special meaning to an HTML browser into characters which can be displayed as intended.	es.tld
<es:counter>	Creates a for loop.	es.tld
<es:date>	Gets a date- and time-formatted String based on the user's time zone preference.	es.tld
<es:forEachInArray>	Iiterate over an array.	es.tld

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<es:isNull>`	Checks to see if a value is null. In the case of a `String`, the `<es:isNull>` tag is used to check if the `String` is null or has a value. An empty string will cause `isNull` to be `false` (an empty string is not null).	`es.tld`
`<es:notNull>`	The `<es:notNull>` tag is used to check if a value is not null. In the case of a `String`, the `<es:notNull>` tag is used to check if the `String` is not null or has a value. An empty string will cause `notNull` to be `true` (an empty string is treated as a value).	`es.tld`
`<es:transposeArray>`	The `<es:transposeArray>` tag is used to transpose a standard [row][column] array to a [column][row] array.	`es.tld`
`<es:uriContent>`	Pulls content from a URL. It is best used for grabbing text-heavy pages.	`es.tld`
`<i18n:getMessage>`	Used in conjunction with the `<i18n:localize>` tag to retrieve localized static text or messages from a JspMessageBundle.	`i18n.tld`
`<i18n:localize>`	Allows you to define the language, country, variant, and base bundle name to be used throughout a page when accessing resource bundles via the `<i18n:getmessage>` tag. .	`i18n.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
<ph:placeholder>	Implements a placeholder, which describes the behavior for a location on a JSP page. You use the E-Business Control Center to define a placeholder. See also <ad:adTarget>.	ph.tld
<portal:createPortalPageChangeURL>	Generates a webflow URL for a page change event.	portal.tld
<portal:createWebflowURL>	Dynamically creates a Webflow URL in a JSP. The Webflow URL may include the protocol, domain name, port, Web application URI, WebflowServlet URI, and query string.	portal.tld
<portal:form>	Generatse an HTML form tag. This tag is not as sophisticated as the <portal:validatedForm> tag, but is simpler. For more information about the <portal:validatedForm> tag, refer to the next section.	portal.tld
<portal:validatedForm>	Dynamically generates HTML forms that can be validated. When a Web site visitor enters invalid information, the visitor's input is preserved and redisplayed with an appropriate error message.	portal.tld
<portlet:createPortletEditURL>	Generates a webflow URL that represents editing a portlet.	portlet.tld
<portlet:createPortletFloatURL>	Generates a webflow URL that represents creating a "floating" portlet, which is a portlet that appears in an independent window.	portlet.tld

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<portlet:createPortletMaximizeURL>`	Generates a webflow URL that represents maximizing a portlet.	`portlet.tld`
`<portlet:createPortletMinimizeURL>`	Generates a webflow URL that represents minimizing a portlet.	`portlet.tld`
`<portlet:createPortletUneditURL>`	Generates a webflow URL that represents leaving the edit URL and moving to another page.	`portlet.tld`
`<portlet:createPortletUnmaximizeURL>`	Generates a webflow URL that represents unmaximizing a portlet.	`portlet.tld`
`<portlet:createPortletUnminimizeURL>`	Generates a webflow URL that represents unminimizing a portlet.	`portlet.tld`
`<portlet:createWebflowURL>`	Dynamically creates a Webflow URL in a JSP. The Webflow URL may include the protocol, domain name, port, Web application URI, `WebflowServlet` URI, and query string.	`portlet.tld`
`<portlet:form>`	Dynamically generates an HTML form tag. This tag is not as sophisticated as the `<portlet:validatedForm>` tag, but is simpler to use.	`portlet.tld`
`<portlet:getException>`	Retrieves the exception or message thrown by a webflow processor. This can be the message associated with a InvalidFormFieldException or ProcessingException. This tag can be inlined in which it calls getMessage() on the exception or return a scripting variable representing the exception itself.	`portlet.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<portlet:validatedForm>`	Dynamically generates HTML forms that can be validated. When a Web site visitor enters invalid information, the visitor's input is preserved and redisplayed with an appropriate error message.	`portlet.tld`
`<ps:getPropertyNames>`	Retrieves a list of property names given a property set.	`ps.tld`
`<ps:getPropertySetNames>`	Retrievest a list of property sets given a property set type.	`ps.tld`
`<ps:getRestrictedPropertyValues>`	Returns a list of restricted values for a specific property definition, converted into Strings. These values will be returned as an array of Strings.	`ps.tld`
`<pz:contentQuery>`	Performs a content attribute search for content in a content manager. If the `useCache` attribute is set to `true`, the results of a content management query will be cached. The tag only has a begin tag and does not have a body or end tag. It returns an array of `Content` objects returned from the content manager as the result of executing the content query.	`cm.tld`
`<pz:contentSelector>`	Allows arbitrary personalized content to be recommended based on a content selector rule.	`cm.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<pz:div>`	Allows a piece of content to be conditionally included as a result of a classifier rule being executed by the rules engine. If the user's profile matches the classification specified in the E-Business Control Center, then the conditional content is included. This tag has a begin tag, a body, and an end tag. The tag returns a list of `Classification` objects that the user belongs to.	`cm.tld`
`<tr:clickContentEvent>`	Generatse a behavior event when a user has clicked (through) on an ad impression. This tag will return a URL query string containing event parameters. It is then used when forming the complete URL that hyperlinks the content	`tracking.tld`
`<tr:displayContentEvent>`	Generates a behavior event when a user has received (viewed) an ad impression, (typically a gif image).	`tracking.tld`
`<trp:clickProductEvent>`	Generates a behavior event when a user has clicked (through) on a product impression. This tag will return a URL query string containing event parameters. It is then used when forming the complete URL that hyperlinks the content.	`productTracking.tld`
`<trp:displayProductEvent>`	Generates a behavior event when a user has received (viewed) a product impression, (typically a gif image). At least one of `sku`, `categoryId`, or `documentId` is required.	`productTracking.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
<um:addGroupToGroup>	Adds the group corresponding to the provided childGroupName to the group corresponding to the provided groupName. Since a group can only have one parent, any previous database records which reflect the group belonging to another parent will be destroyed. Both the parent group and the child group must previously exist for proper tag behavior. The tag has no enclosed body.	um.tld
<um:addUserToGroup>	Adds the user corresponding to the provided username to the group corresponding to the provided groupName. Both the specified user and the specified group must previously exist for proper tag behavior. The tag has no enclosed body.	um.tld
<um:createGroup>	Creates a new group in the realm, and a corresponding group profile in the personalization database. This tag has no enclosed	um.tld
<um:createUser>	Creates a new user profile. This tag has no enclosed body. Although classified as a Group-User management tag, this tag can be used in conjunction with run-time activities, in that it will persist any properties associated with a current Anonymous User Profile if specified.	um.tld
<um:getChildGroupNames>	Returns the names of any groups that are children of the given group.	um.tld

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<um:getGroupNamesForUser>`	Retrieves a `String` array that contains the group names corresponding to groups to which the provided user immediately belongs. This tag has no enclosed body.	`um.tld`
`<um:getParentGroupName>`	Retrieves the name of the parent of the group associated with the provided `groupName`. The information is taken from the realm. This tag has no enclosed body.	`um.tld`
`<um:getProfile>`	Retrieves the profile corresponding to the provided profile key and profile type. The tag has no enclosed body. The retrieved profile can be treated as a `com.bea.p13n.usermgmt.profile.ProfileWrapper`.	`um.tld`
`<um:getProperty>`	Retrieves the property value for a specified property set-property name pair. The tag has no enclosed body. The value returned is an `Object`.	`um.tld`
`<um:getPropertyAsString>`	Works exactly like the `<um:getProperty>` tag, but ensures that the retrieved property value is a `String`. The following example shows a multi-valued property which returns a `Collection`, but presents a list of favorite colors.	`um.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
<um:getTopLevelGroups>	Retrieves an array of group names, each of which has no parent group. The information is taken from the realm. This tag has no enclosed body.	um.tld
<um:getUsernames>	Retrieves a String array that contains the usernames matching the provided search expression. The search expression supports only the asterisk (*) wildcard character, and is case insensitive. As many asterisks as desired may be used in the search expression. This tag has no enclosed body.	um.tld
<um:getUsernamesForGroup>	Retrieves a String array that contains the usernames matching the provided search expression and correspond to members of the provided group. The search expression supports only the asterisk (*) wildcard character, and is case insensitive. As many asterisks as desired may be used in the search expression. This tag has no enclosed body.	um.tld
<um:login>	Provides weak authentication (username, password) against the current security realm, and sets the authenticated user as the current WebLogic user. This tag has no enclosed body.	um.tld
<um:logout>	Ends the current user's WebLogic Server session. This tag should be used in combination with the <um:login> tag.	um.tld

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<um:removeGroup>`	Removes the group corresponding to the provided `groupName`. This tag has no enclosed body.	`um.tld`
`<um:removeGroupFromGroup>`	Removes a child group from a parent group.	`um.tld`
`<um:removeProperty>`	Removes the specified property from the current session's profile or from the Anonymous User Profile. The tag has no enclosed body. Subsequent calls to `<um:getProperty>` for a removed property would result in the default value for the property as prescribed by the property set, or from the Profile's successor.	`um.tld`
`<um:removeUser>`	Removes the user corresponding to the provided `username`. It can remove any type of extended user that has its profileType set in the database. This tag has no enclosed body.	`um.tld`
`<um:removeUserFromGroup>`	Removes a user from a group. **This tag should only be invoked when the current realm is an implementation of `weblogic.security.acl. ManageableRealm`. This interface is implemented by the default WebLogic Portal realm (`com.bea.p13n.security.realm. RDBMSRealm`).	`um.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
<um:setPassword>	Updates the password for the user corresponding to the provided username.**This tag should only be invoked when the current realm is an implementation of weblogic.security.acl. ManageableRealm. .	um.tld
<um:setProperty>	Updates a property value for either the session's current profile, or for the Anonymous User Profile. This tag has no enclosed body.	um.tld
<util:invalidURL>	Processes the body if the supplied url is null or less than four characters long.	util.tld
<util:validURL>	Processes the body if the supplied url is not null and greater than three characters long.	util.tld
<webflow:checkbox>	Generates the required HTML for a check box. This tag will preserve input upon the InputProcessor throwing an InvalidFormFieldException. This tag must be nested in the <webflow:validatedForm> tag.	webflow.tld
<webflow:createResourceURL>	Creates a static URL for a resource, using the value of the P13N_STATIC_ROOT context parameter in the application's WEB-INF/web.xml file. This tag may be used to load GIF images from a separate server.	webflow.tld

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<webflow:createWebflowURL>`	Dynamically creates a Webflow URL in a JSP. The Webflow URL may include the protocol, domain name, port, Web application URI, `WebflowServlet` URI, and query string.	`webflow.tld`
`<webflow:form>`	Dynamically generates an HTML form tag. This tag is not as sophisticated as the `<webflow:validatedForm>` tag, but is simpler.	`webflow.tld`
`<webflow:getException>`	Retrieves the exception or message thrown by a Webflow processor. This can be the message associated with an `InvalidFormFieldException` exception or a `ProcessingException` exception. This tag can be inlined (in which it calls the `getMessage()` method on the exception) or return a scripting variable representing the exception.	`webflow.tld`
`<webflow:getProperty>`	Retrieves a named property from the Pipeline Session. This tag can be inlined or return a scripting variable.	`webflow.tld`
`<webflow:getValidatedValue>`	Display the fields in a form that a Web site visitor must correct. The `<webflow:getValidatedValue>` tag is used in tandem with the`<webflow:setValidatedValue>` tag.	`webflow.tld`

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
<webflow:option>	Represent an HTML option, but preserves the input. An option is one value in a list box. This tag must be nested in the <webflow:select> tag.	webflow.tld
<webflow:password>	This tag is similar to the <webflow:text> tag except that field input is masked with asterisks (****). This tag must be nested in the <webflow:validatedForm> tag.	webflow.tld
<webflow:radio>	Represents an HTML radio button, but preserves the input. This tag must be nested in the <webflow:validatedForm> tag.	webflow.tld
<webflow:select>	Represent a list box, but preserves its input. This tag must be nested in the <webflow:validatedForm> tag.	webflow.tld
<webflow:setProperty>	Sets a property in the Pipeline Session.	webflow.tld
<webflow:setValidatedValue>	Configures the display of fields in a form that a Web site visitor must correct. Usually this is done within an Input Processor, but it can also be done from a JSP by using this tag.	webflow.tld
<webflow:text>	Validates an HTML text field. This tag must be nested in the <webflow:validatedForm> tag.	webflow.tld

Table C-1 Portal Development JSP Tags by Name (Continued)

Tag Name	Description	Tag Library
`<webflow:textarea>`	Represents an HTML text area, but preserves the input. This tag must be nested in the `<webflow:validatedForm>` tag.	`webflow.tld`
`<webflow:validatedForm>`	Dynamically creates the URL that defines the action in an HTML form. This tag should be used in conjunction with the `com.bea.p13n.appflow.webflow.forms.*` package and the other nested form tags defined in the `webflow.tld` file.	`webflow.tld`
`<wl:cache>`	Specifies that its contents do not necessarily need to be updated every time it is displayed.	`wl.tld`
`<wl:process>`	Queries attribute-based flow control. By using a combination of the four attributes, you can selectively execute the statements between the `<wl:process>` and `</wl:process>` tags.	`wl.tld`
`<wl:repeat>`	Iterates over a variety of Java objects, as specified in the set attribute.	`wl.tld`