



JAVASERVER PAGES™ (JSP™) SYNTAX version 2.0

	Usage	Description	Syntax	
			JSP	XML
		<p>Legend</p> <p>All tags are case sensitive. A pair of single quotes is equivalent to a pair of double quotes. Spaces are not allowed between an equals sign and an attribute value.</p> <p>plain text = required bold = default <i>italics</i> = user-defined = or ' ' = literal ^ = anything other than [] = optional { } = required choice ... = list of items + = can repeat () = grouped together - = a range</p> <p>Notes</p> <p>Some action elements have attributes whose value can be computed at request time. In JSP syntax, the format of such a value is the same as a JSP expression: <code><%= expression %></code>. In XML syntax, the format of the value is <code>%= expression %</code>. When using the JSP expression language (EL), the format of the value in either syntax is <code>\${ expression }</code>.</p> <p>EL expressions can be quoted by adding a backslash to the front of the expression when the expression language is enabled. This prevents the expression from being evaluated. Example: <code>\\${x+y}</code> will display as <code>\${x+y}</code>.</p> <p>All elements in XML syntax can contain an <code>xmlns</code> attribute that complies with the XML namespaces spec. This attribute is used to declare tag libraries and other namespaces. Within the start tag of an element, its syntax is <code>[xmlns:taglibprefix="URI"]+</code></p> <p>All JSP standard actions and custom actions can contain a <code>jsp:attribute</code> standard element as a substitute for any of its attributes. The <code>jsp:attribute</code> syntax is <code><jsp:attribute name=<i>attributeName</i> [trim="true false"] /></code> If an action contains any <code>jsp:attribute</code> elements and the action also has a body, it must use the <code>jsp:body</code> tag to represent the body. The <code>jsp:body</code> syntax is <code><jsp:body> any elements or text </jsp:body></code></p> <p>The Usage column indicates where the element can be used. S = JSP pages. X= JSP documents. TS= tag files in standard syntax. TX= tag files in XML syntax. A = All.</p>		
Comment	A	Documents the JSP file, but is not included in the response. The comment can include anything except <code>--></code> (or <code>--></code> if the comment is in XML syntax).	<code><%-- comment --%></code>	<code><!-- comment --></code>
Declaration	A	Declares variables or methods valid in the page's scripting language.	<code><%! declaration; [declaration;]+ ... %></code>	<code><jsp:declaration> code fragment [declaration;]+ ... </jsp:declaration></code>
EL Expression	A	Contains an expression in the JSP Expression Language (EL). See Expression Language section below for the syntax.	<code>\${ Expression }</code>	same as JSP syntax

JAVASERVER PAGES™ (JSP™) SYNTAX version 2.0

	Usage	Description	Syntax	
			JSP	XML
Expression	A	Contains an expression valid in the page's scripting language.	<%= <i>expression</i> %>	<jsp:expression> <i>expression</i> </jsp:expression>
Scriptlet	A	Contains a code fragment valid in the page's scripting language.	<% <i>code fragment</i> %>	<jsp:scriptlet> <i>code fragment</i> </jsp:scriptlet>

Directives	Usage	Description	Syntax	
			JSP	XML
Attribute	TS, TX	Declares an attribute of the custom tag defined in the tag file.	<%@ attribute name=" <i>attribute-name</i> " [required="true false"] [fragment="true false"] [rtexprvalue="true false"] [type="java.lang.String a non-primitive type"] [description="text"] %>	<jsp:directive.attribute <i>attributeDirectiveAttrList</i> /> where <i>attributeDirectiveAttrList</i> is the same as the list in the JSP column.
Include	A	Includes a file, parsing the file's JSP elements.	<%@ include file=" <i>relativeURL</i> " %>	<jsp:directive.include file=" <i>relativeUR</i> " />
Page	S, X	Defines attributes that apply to a JSP page.	<%@ page [language="java"] [extends="package.class"] [import="{package.class package.*} , ... "] [session="true false"] [buffer="none 8kb sizekb"] [autoFlush="true false"] [isThreadSafe="true false"] [info="text"] [errorPage=" <i>relativeURL</i> "] [isErrorPage="true false"] [contentType="{mimeType [; charset=characterSet] text/html ; charset=ISO-8859-1}"] [pageEncoding="{characterSet ISO-8859-1}"] [isELIgnored="true false"] %>	<jsp:directive.page <i>pageDirectiveAttrList</i> /> where <i>pageDirectiveAttrList</i> is the same as the list in the JSP column.

JAVASERVER PAGES™ (JSP™) SYNTAX version 2.0

Directives	Usage	Description	Syntax	
			JSP	XML
Tag	TS, TX	<p>Similar to the page directive in a JSP page, but applies to tag files instead of JSP pages. As with the page directive, a translation unit can contain more than one instance of the tag directive. All the attributes apply to the complete translation unit. However, there can be only one occurrence of any attribute or value defined by this directive in a given translation unit. With the exception of the import attribute, multiple attribute or value (re)definitions result in a translation error.</p> <p>Also used for declaring custom tag properties such as display name.</p>	<pre><%@ tag [display-name="<i>name of the tag file</i> <i>display-name</i>"] [body-content="scriptless tagdependent empty"] [dynamic-attributes="<i>page-scoped attribute</i>"] [small-icon="<i>relativeURL</i>"] [large-icon="<i>relativeURL</i>"] [description="<i>text</i>"] [example="<i>text</i>"] [language="java"] [import="{<i>package.class</i> <i>package.*</i>}, ... "] [pageEncoding="{<i>characterSet</i> ISO-8859-1"}] [isELIgnored="true false"] %></pre>	<pre><jsp:directive.tag <i>tagDirectiveAttrList</i> /></pre> <p>where <i>tagDirectiveAttrList</i> is the same as the list in the JSP column.</p>
Taglib	S, TS	<p>Defines a tag library and prefix for custom tags used in the JSP page.</p>	<pre><%@ taglib {uri="<i>URI</i>" tagdir="/WEB-INF/tags[/subdir]+" } prefix="<i>tagPrefix</i>" %></pre>	<p>No equivalent. Included in <jsp:root> and in any other XML element using the xmlns attribute instead:</p> <pre><anyxmlelement xmlns:prefix="{<i>uri</i> urn:jsptld:<i>path</i> urn:jsptagdir:/WEB-INF/tags[/subdir]+ }" ></pre> <p>Examples:</p> <pre><my:tag xmlns:my="http://example.com/mytaglibrary" /> <my:tag xmlns:my="urn:jsptld:/WEB-INF/tlds/my.tld" /> <my:tag xmlns:my="urn:jsptagdir:/WEB-INF/tags" /></pre>
Variable	TS, TX	<p>Declares an expression language variable exposed by the tag to the calling page.</p>	<pre><%@ variable { name-given="<i>scripting variable</i>" (name-from-attribute="<i>scripting variable</i>" alias="<i>locally-scoped attribute</i>") } [variable-class="java.lang.String <i>name of the variable class</i>"]] [declare="true false"] [scope="AT_BEGIN AT_END NESTED"] [description="<i>text</i>"] %></pre>	<pre><jsp:directive.variable <i>variableDirectiveAttrList</i> /></pre> <p>where <i>variableDirectiveAttrList</i> is the same as the list in the JSP column.</p>

JAVASERVER PAGES™ (JSP™) SYNTAX version 2.0

Expression Language	Usage	Description	Syntax	
			JSP	XML
ELExpression	A	Allows easy access to application data stored in JavaBeans components. Examples: <a:tag value="{x+y}" /> <a:tag value="{first} {last}" /> <a:tag>{x+y}</a:tag>	{' Expression '}	Same as JSP syntax
Expression	A	Either a choice, binary or unary expression, or a value	{ (ChoiceExpression (Expression BinaryOp Expression) (UnaryOp Expression) Value) }	Same as JSP syntax
ChoiceExpression	A	If first expression evaluates to true, evaluate the second expression; otherwise, evaluate the third expression.	Expression ? Expression : Expression	Same as JSP syntax
BinaryOp	A	Supported arithmetic, logical, and relational operators.	and && or '+' '-' * / div % mod > gt < lt >= ge <= le == eq != ne	and && or '+' '-' * / div % mod gt lt ge le == eq ne
UnaryOp	A	Supported unary operators	{ '-' ! not empty }	Same as in JSP syntax
Value	A	Expresses variables, for example: sessionScope.cart.numOfItems param['mycom.productId']	{ ValuePrefix (Value ValueSuffix) }	Same as JSP syntax
ValuePrefix	A	The prefix of the variable expression. For a list of implicit objects, see the Implicit Objects for EL Expressions table at the end of this card.	{ Literal '(' Expression ')' ImplicitObject Java language identifier FuncInvocation }	Same as JSP syntax
ValueSuffix	A	The suffix of the variable expression. The . operator is automatically evaluated into the equivalent [] operator. More specifically, expr-a.identifier-b is equivalent to expr-a["identifier-b"].	{ Identifier '(' Expression ')' }	Same as JSP syntax

Expression Language	Usage	Description	Syntax	
			JSP	XML
FuncInvocation	A	Defines a function that can be invoked in an expression. For example: f:equals(A, B) where f is the prefix declared in a taglib directive importing the tag library that defines the function.	[Identifier :] Identifier '(' [Expression [, ' Expression]+] ')'	Same as JSP syntax
Literal	A	Supported literals: boolean, integer, floating point, String, and null.	{true false} (0-9)+ FloatingPtLiteral StringLiteral null	Same as JSP syntax
FloatingPtLiteral	A	Represents a floating-point literal.	[0-9]+ . (0-9)+ [Exponent] (0-9)+ [Exponent]	Same as JSP syntax
Exponent	A	Represents an exponent used in FloatingPtLiteral	{e E} ['+' '-'] (0-9)+	Same as JSP syntax
StringLiteral	A	A string of any characters enclosed by single or double quotes: 'string' "string" It is possible to use single quotes within double quotes, and vice versa, without escaping. Escape characters: \\ = Escaped \ \' = Escaped ', valid only inside " \" = Escaped ", valid only inside '	{['^(' \) \' \\]+ "[^(" \) \' \\]+}	Same as JSP syntax. XML syntax rules must be followed with respect to quoting inside attribute values.

Standard Elements	Usage	Description	Syntax	
			JSP	XML
<jsp:element>	A	Dynamically generates an element.	(<jsp:element name=" <i>elementName</i> " (/> > (<i>any elements or text</i> </jsp:element>)) (<jsp:element name=" <i>elementName</i> "> [<jsp:attribute name=" <i>attributeName</i> " [trim=" true false"] (/> (<i>any elements or text</i> </jsp:attribute>))]+ [<jsp:body> <i>any elements or text</i> </jsp:body>] </jsp:element>)	Same as JSP syntax

Standard Elements	Usage	Description	Syntax	
			JSP	XML
<jsp:doBody>	T	Evaluates the body of the tag used by the calling page to invoke this tag file.	<jsp:doBody ({var="scopedAttributeName" varReader="scopedAttributeName" } [scope="page request session application"] />) />	Same as JSP syntax.
<jsp:forward>	A	Forwards a request to a web resource.	<jsp:forward page="{relativeURL '{ Expression }'" <%= expression %>" { /> > [<jsp:param name="parameterName" value="{parameterValue '{ Expression }'" <%= expression %>" } />] + </jsp:forward> }	<jsp:forward page="{relativeURL '{ Expression }'" <%= expression %>" { /> > [<jsp:param name="parameterName" value="{parameterValue '{ Expression }'" <%= expression %>" } />] + </jsp:forward> }
<jsp:getProperty>	A	Inserts the value of a bean property into the result.	<jsp:getProperty name="beanInstanceName" property="propertyName" />	Same as JSP syntax.
<jsp:include>	A	Includes a static file or the result from another web component.	<jsp:include page="{relativeURL '{ Expression }'" <%= expression %>" [flush="true false"] { /> > [<jsp:param name="parameterName" value="{parameterValue '{ Expression }'" <%= expression %>" } />] +	<jsp:include page="{relativeURL '{ Expression }'" <%= expression %>" [flush="true false"] { /> > [<jsp:param name="parameterName" value="{parameterValue '{ Expression }'" <%= expression %>" } />] +
<jsp:invoke>	TS, TX	Evaluates a fragment attribute.	<jsp:invoke fragment="fragmentName" ({var="scopedAttributeName" varReader="scopedAttributeName" } [scope="page request session application"] />) />	Same as JSP syntax.
<jsp:output>	X, TX	Specifies the XML declaration or the document type declaration in the request output of a JSP document or a tag file that is in XML syntax.	No equivalent.	<jsp:output (omit-xml-declaration="yes no true false") { doctypeDecl } /> doctypeDecl ::= (doctype-root-element="rootElement" doctype-public="PubidLiteral" doctype-system="SystemLiteral") (doctype-root-element="rootElement" doctype-system="SystemLiteral")

Standard Elements	Usage	Description	Syntax	
			JSP	XML
<jsp:plugin>	A	Causes the execution of an applet or bean. The applet or bean executes in the specified plugin. If the plugin is not available, displays a dialog to initiate the download of the plugin software.	<pre><jsp:plugin type="bean applet" code="classFileName" codebase="classFileDirectoryName" [name="instanceName"] [archive="URIToArchive, ..."] [align="bottom top middle left right"] [height="{displayPixels '{\$ Expression }' / <%= expression %>}"] [width="{displayPixels '{\$ Expression }' <%= expression %>}"] [hspace="leftRightPixels"] [vspace="topBottomPixels"] [jreversion="JREVersionNumber 1.2"] [nspluginurl="URLToPlugin"] [iepluginurl="URLToPlugin"] [mayscript="true false"> [<jsp:params> [<jsp:param name="parameterName" value="{parameterValue '{\$ Expression }' <%= expression %>}" />]+ </jsp:params>] [<jsp:fallback> text message if plugin download fails </jsp:fallback>] </jsp:plugin></pre>	<pre><jsp:plugin type="bean applet" code="classFileName" codebase="classFileDirectoryName" [name="instanceName"] [archive="URIToArchive, ..."] [align="bottom top middle left right"] [height="{displayPixels '{\$ Expression }' %= expression %}"] [width="{displayPixels '{\$ Expression }' %= expression %}"] [hspace="leftRightPixels"] [vspace="topBottomPixels"] [jreversion="JREVersionNumber 1.2"] [nspluginurl="URLToPlugin"] [iepluginurl="URLToPlugin"] > [<jsp:params> [<jsp:param name="parameterName" value="{parameterValue '{\$ Expression }' %= expression %}" />]+ </jsp:params>] [<jsp:fallback> text message if plugin download fails </jsp:fallback>] </jsp:plugin></pre>
<jsp:root>	X, TX	<p>Must represent a root element and defines standard elements and namespace attributes of tag libraries. Identifies the JSP page as a JSP document to the container.</p> <p>This element is optional in JSP documents written in version 2.0 of JSP syntax. In version 2.0 JSP documents, any element can be a root element, and any element can declare namespaces using xmlns.</p>	No equivalent.	<pre><jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" [xmlns:taglibPrefix="URI"]+ ... version="2.0"> other elements </jsp:root></pre>
<jsp:setProperty>	A	Sets a bean property value or values.	<pre><jsp:setProperty name="beanInstanceName" { property="*" property="propertyName" [param="parameterName"] property="propertyName" value="{string '{\$ Expression }' <%= expression %>}" } /></pre>	<pre><jsp:setProperty name="beanInstanceName" { property="*" property="propertyName" [param="parameterName"] property="propertyName" value="{string '{\$ Expression }' %= expression %}" } /></pre>

Standard Elements	Usage	Description	Syntax	
			JSP	XML
<jsp:text>	A	Encloses template data, which is passed through uninterpreted to the current output while preserving whitespace.	<pre><jsp:text> template data </jsp:text></pre>	Same as JSP syntax. When included in a JSP document, the template data must be well-formed XML.
<jsp:useBean>	A	Locates or instantiates a bean with a specific name and scope.	<pre><jsp:useBean id="beanInstanceName" scope="page request session application" { class="package.class" [type="package.class"] beanName="{package.class '{\$ Expression }'}" <%= expression %>" type="package.class" type="package.class" } { /> > other elements </jsp:useBean> }</pre>	<pre><jsp:useBean id="beanInstanceName" scope="page request session application" { class="package.class" [type="package.class"] beanName="{package.class '{\$ Expression }'}" %= expression %}" type="package.class" type="package.class" } { /> > other elements </jsp:useBean> }</pre>
<tagPrefix:name>		Accesses a custom tag's functionality.	<pre><tagPrefix:name attribute="value"+ ... /> <tagPrefix:name attribute="value"+ ... > other tags and data </tagPrefix:name></pre>	<pre><tagPrefix:name attribute="value"+ ... /> <tagPrefix:name attribute="value"+ ... > other tags and data </tagPrefix:name></pre>

Implicit Objects for Scriptlets	Type	Scope	Some Useful Methods (see class or interface for others)
request	Subclass of <i>javax.servlet.ServletRequest</i>	Request	getAttribute, getParameter, getParameterNames, getParameterValues, setAttribute
response	Subclass of <i>javax.servlet.ServletResponse</i>	Page	Not typically used by JSP page authors
pageContext	<i>javax.servlet.jsp.PageContext</i>	Page	findAttribute, getAttribute, getAttributesScope, getAttributeNamesInScope, setAttribute
session	<i>javax.servlet.http.HttpSession</i>	Session	getAttribute, getId, setAttribute
application	<i>javax.servlet.ServletContext</i>	Application	getAttribute, getMimeType, getRealPath, setAttribute
out	<i>javax.servlet.jsp.JspWriter</i>	Page	clear, clearBuffer, flush, getBufferSize, getRemaining
config	<i>javax.servlet.ServletConfig</i>	Page	getInitParameter, getInitParameterNames
page	<i>java.lang.Object</i>	Page	Not typically used by JSP page authors
exception	<i>java.lang.Throwable</i>	Page	getMessage, getLocalizedMessage, printStackTrace, toString

Implicit Objects for EL Expressions	Type	Description
<code>pageContext</code>	<i>javax.servlet.jsp.PageContext</i>	The context for the JSP page. Provides access to various objects, including <code>servletContext</code> , <code>session</code> , <code>request</code> , and <code>response</code> .
<code>pageScope</code>	<i>java.util.Map</i>	Maps page-scoped variable names to their values.
<code>requestScope</code>	<i>java.util.Map</i>	Maps request-scoped variable names to their values.
<code>sessionScope</code>	<i>java.util.Map</i>	Maps session-scoped variable names to their values.
<code>applicationScope</code>	<i>java.util.Map</i>	Maps application-scoped variable names to their values.
<code>param</code>	<i>java.util.Map</i>	Maps a request parameter to a single String parameter value (obtained by calling <code>ServletRequest.getParameter(String name)</code>).
<code>paramValues</code>	<i>java.util.Map</i>	Maps a request parameter name to an array of String values for that parameter name (obtained by calling <code>ServletRequest.getParameterValues(String name)</code>).
<code>header</code>	<i>java.util.Map</i>	Maps a request header name to a single String header value (obtained by calling <code>ServletRequest.getHeader(String name)</code>).
<code>headerValues</code>	<i>java.util.Map</i>	Maps a request header name to an array of String values for that header (obtained by calling <code>ServletRequest.getHeaders(String)</code>).
<code>cookie</code>	<i>java.util.Map</i>	Maps a cookie name to a single Cookie object. Cookies are retrieved according to the semantics of <code>HttpServletRequest.getCookies()</code> . If the same name is shared by multiple cookies, an implementation must use the first one encountered in the array of Cookie objects returned by the <code>getCookies()</code> method. However, the ordering of cookies is currently unspecified in the Servlet specification.
<code>initParam</code>	<i>java.util.Map</i>	Maps a context initialization parameter name to a String parameter value (obtained by calling <code>ServletContext.getInitparameter(String name)</code>).