**JSP Expressions**

A **JSP expression** is used to insert the value of a scripting language expression, converted into a string, into the data stream returned to the client. When the scripting language is the Java programming language, an expression is transformed into a statement that converts the value of the expression into a String object and inserts it into the implicit out object.

The syntax for an expression is as follows:

<%= *scripting-language-expression* %>

Note that a semicolon is not allowed within a JSP expression, even if the same expression has a semicolon when you use it within a scriptlet.

The code placed within the **JSP expression tag** is *written to the output stream of the response*. **So you need not write out.print() to write data**. It is mainly used to print the values of the variable or method. Expression always becomes the argument to a print() method.

**<**%=  statement %**>**

**<html>**

**<body>**

**<**%= "welcome to jsp" %**>**

**</body>**

**</html>**

**<html>**

**<body>**

Current Time: **<**%= java.util.Calendar.getInstance().getTime() %**>**

**</body>**

**</html>**

In this example, we are printing the username using the expression tag. The index.html file gets the username and sends the request to the welcome.jsp file, which displays the username.

*File: index.jsp*

**<html>**

**<body>**

**<form** action="welcome.jsp"**>**

**<input** type="text" name="uname"**><br/>**

**<input** type="submit" value="go"**>**

**</form>**

**</body>**

**</html>**

*File: welcome.jsp*

**<html>**

**<body>**

**<**%= "Welcome "+request.getParameter("uname") %**>**

**</body>**

**</html>**

To simplify these expressions, you can use a number of **predefined variables** (or “implicit objects”). There is nothing magic about these variables; the system simply tells you what names it will use for the local variables in \_jspService (the method that replaces doGet in servlets that result from JSP pages). The most important ones are these:

• **request**, the HttpServletRequest.

• **response**, the HttpServletResponse.

• **session**, the HttpSession associated with the request (unless disabled with the session attribute of the page directive).

• **out**, the Writer (a buffered version of type JspWriter) used to send output to the client.

• **application**, the ServletContext. This is a data structure shared by all servlets and JSP pages in the Web application and is good for storing shared data.

Here is an example:

Your hostname: <%= **request**.getRemoteHost() %>

JSP expressions basically become print (or write) statements in the servlet that results from the JSP page. Whereas regular HTML becomes print statements with double quotes around the text, JSP expressions become print statements with no double quotes. Instead of being placed in the doGet method, these print statements are placed in a new method called \_jspService that is called by service for both GET and POST requests.

If you want to see the exact code that your server generates, you’ll have to dig around a bit to find it. In fact, some servers delete the source code files once they are successfully compiled.

Tomcat Autogenerated Servlet Source Code in install\_dir/**work**/... The location varies slightly among various Tomcat versions.)

Expressions.jsp

<HTML>

<HEAD>

<TITLE>JSP Expressions</TITLE>

<META NAME="keywords"

CONTENT="JSP,expressions,JavaServer Pages,servlets">

<META NAME="description"

CONTENT="A quick example of JSP expressions.">

<LINK REL=STYLESHEET

HREF="JSP-Styles.css"

TYPE="text/css">

</HEAD>

<BODY>

<H2>JSP Expressions</H2>

<UL>

<LI>Current time: <%= new java.util.Date() %>

<LI>Server: <%= application.getServerInfo() %>

<LI>Session ID: <%= session.getId() %>

<LI>The <CODE>testParam</CODE> form parameter:

<%= request.getParameter("testParam") %>

</UL>

</BODY></HTML>

Any processing requires using a **declaration** instead of an expression.