Notes on JSP

JSP is a text documents that holds two things: Static elements and JSP elements.  
The static elements are just things like HTMl, CSS, PHP, etc.  
The JSP elements are the dynamic elements that intermingle with the static elements to produce unique pages.   
So you would have something like

<html>…blahblah… JSP ELEMENT THING ...blahblah…</html>

The JSP elemnt thing is dynamic and can be anything, for instance an entire java class system to animate birds.

contentType Directive

Directives are sort of like pre-processor definitions that tell the JSP to do something before running its code.

How do you choose what the static type of a JSP is? Use this directive flag.

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

It’s what makes the .JSP know that html is being used, and it should be parsed accordingly.

Page Directive

Used to import things like STL. So you can import java.util using tags like:

<%@ page import="java.util.\*,java.text.\*" %>

Include Directive

Used to import other files (other JSPs).

<%@ include file="hello.jsp" %>

Evaluating

${variable}

Is the simplest evaluation, and will return the value stored in the component called variable.

Expressions

To add dynamic content to a JSP that will be evaluated at run-time, use the tag symbols: <%= and %>. So,

<%= new java.util.Date() %>

Will evaluate the date and insert it there at run time.

Scriplets

This is the guts and core of the JSP. To insert actual java that will run in your JSP dynamic sections, you use the tags <% and %>. These are called scriplets.   
You can create local variables, run code, created classes, and do generally JAVA things in blocks of scriplets.

Mixing scriplets and static content

Nothing better than an example, here. Just see that you can intertwine the static content with Java code to create dynamic and highly efficient static code. In this example, HTML:

<TABLE BORDER=2>

<%

    for ( int i = 0; i < n; i++ ) {

        %>

        <TR>

        <TD>Number</TD>

        <TD><%= i+1 %></TD>

        </TR>

        <%

    }

%>

</TABLE>

tagLib

tagLib is basically an import statement to import a bunch of tags that can then be used:

<%@ taglib prefix="tt" [tagdir=/WEB-INF/tags/*dir* | uri=*URI* ] %>

<%@ taglib prefix="ff" [tagdir=/WEB-INF/tags/*dir2* | uri=*URI* ] %>

Prefix allows you to distinguish tags from one tagLib to another, even if the tags have the same name. It acts like a scope operator.

Creating methods within JSPs

Use the <%! And %> tags to indicate that there will be methods in the scriplet that can then be used in other scriplets and expressions.

<%@ page import="java.util.\*" %>

<HTML>

<BODY>

<%!

    Date theDate = new Date();

    Date getDate()

    {

        System.out.println( "In getDate() method" );

        return theDate;

    }

%>

Hello!  The time is now <%= getDate() %>

</BODY>

</HTML>

**HARD NOTE:** JSPs run on multiple threads. Don’t use variables between scriplets if you can help it, that will end poorly for you because of the multi-threading nature of JSPs.

Tags

JSP tags act like HTML tags. They use < and /> to start and end a tag. Colons are used as scope. You can also combine the start and end tag into a single ender tag if it doesn’t need a body:

<some:tag> … </some:tag>

<some:tag/>

Tags can come from two places: External libraries you import, or predefined tags. Predefined tags start with ‘jsp.’ Two useful predefined tags are jsp:include and jsp:forward.

<HTML>

<BODY>

Going to include hello.jsp...<BR>

<jsp:include page="hello.jsp"/>

<jsp:forward page="hello.jsp"/>

</BODY>

</HTML>

Sessions

On a typical web site, a visitor might visit several pages and perform several interactions.

If you are programming the site, it is very helpful to be able to associate some data with each visitor.  For this purpose, "session"s can be used in JSP.

A session is an object associated with a visitor.  Data can be put in the session and retrieved from it, much like a Hashtable.  A different set of data is kept for each visitor to the site.

<HTML>

<BODY>

<FORM METHOD=POST ACTION="SaveName.jsp">

What's your name? <INPUT TYPE=TEXT NAME=username SIZE=20>

<P><INPUT TYPE=SUBMIT>

</FORM>

</BODY>

</HTML>

The target of the form is "SaveName.jsp", which saves the user's name in the session.  Note the variable "session".  This is another variable that is normally made available in JSPs, just like out and request variables.  (In the @page directive, you can indicate that you do not need sessions, in which case the "session" variable will not be made available.)

<%

   String name = request.getParameter( "username" );

   session.setAttribute( "theName", name );

%>

<HTML>

<BODY>

<A HREF="NextPage.jsp">Continue</A>

</BODY>

</HTML>

The SaveName.jsp saves the user's name in the session, and puts a link to another page, NextPage.jsp.

NextPage.jsp shows how to retrieve the saved name.

<HTML>

<BODY>

Hello, <%= session.getAttribute( "theName" ) %>

</BODY>

</HTML>

If you bring up two different browsers (not different windows of the same browser), or run two browsers from two different machines, you can put one name in one browser and another name in another browser, and both names will be kept track of.

The session is kept around until a timeout period.  Then it is assumed the user is no longer visiting the site, and the session is discarded.