Controlling the Structure of Generated Servlets: The JSP page Directive


Customized Java EE Training: http://courses.coreservlets.com/
Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Hadoop, Android.
Developed and taught by well-known author and developer. At public venues or onsite at your location.

For live Java EE training, please see training courses at http://courses.coreservlets.com/.
JSF 2, PrimeFaces, Servlets, JSP, Ajax (with jQuery), GWT, Android development, Java 6 and 7 programming, SOAP-based and RESTful Web Services, Spring, Hibernate/JPA, XML, Hadoop, and customized combinations of topics.

Taught by the author of Core Servlets and JSP, More Servlets and JSP, and this tutorial. Available at public venues, or customized versions can be held on-site at your organization. Contact hall@coreservlets.com for details.
Agenda

• Understanding the purpose of the page directive
• Designating which classes are imported
• Specifying the MIME type of the page
• Generating Excel spreadsheets
• Participating in sessions
• Setting the size and behavior of the output buffer
• Designating pages to handle JSP errors
• Controlling threading behavior

Purpose of the page Directive

• Give high-level information about the servlet that will result from the JSP page
• Can control
  – Which classes are imported
  – What class the servlet extends
  – What MIME type is generated
  – How multithreading is handled
  – If the servlet participates in sessions
  – The size and behavior of the output buffer
  – What page handles unexpected errors
The import Attribute

• **Format**
  – `<%@ page import="package.class" %>`
  – `<%@ page import="package.class1,...,package.classN" %>`

• **Purpose**
  – Generate import statements at top of servlet definition

• **Notes**
  – Although JSP pages can be almost anywhere on server, classes used by JSP pages must be in normal servlet dirs
  – E.g.:
    – .../WEB-INF/classes or ...
    – .../WEB-INF/classes/directoryMatchingPackage
  – *Always* use packages for utilities that will be used by JSP!

The Importance of Using Packages

• **What package will the system think that SomeHelperClass and SomeUtilityClass are in?**

```java
... public class SomeClass {
  public String someMethod(...) {
    SomeHelperClass test = new SomeHelperClass(...);
    String someString =
      SomeUtilityClass.someStaticMethod(...);
    ...
  }
}
```
The Importance of Using Packages (Continued)

- What package will the system think that SomeHelperClass and SomeUtilityClass are in?

```java
...<%
    SomeHelperClass test = new SomeHelperClass(...);
    String someString =
        SomeUtilityClass.someStaticMethod(...);
%>
```

The import Attribute: Example (Code)

```java
...<h2>The import Attribute</h2>
<%@ page import="java.util.*,coreservlets.*" %>
<%! private String randomID() {
    int num = (int)(Math.random()*1000000.0);
    return("id" + num);
}
private final String NO_VALUE = "<i>No Value</i>";
%>
<% String oldID =
    CookieUtilities.getCookieValue(request, "userID", NO_VALUE);
if (oldID.equals(NO_VALUE)) {
    String newID = randomID();
    Cookie cookie = new LongLivedCookie("userID", newID);
    response.addCookie(cookie);
}
%>
This page was accessed on <%= new Date() %> with a userID cookie of <%= oldID %>. 
</body></html>
```
The import Attribute: Example (Results)

The contentType and pageEncoding Attributes

- **Format**
  - `<%@ page contentType="MIME-Type" %>`
  - `<%@ page contentType="MIME-Type; charset=Character-Set" %>`
  - `<%@ page pageEncoding="Character-Set" %>`

- **Purpose**
  - Specify the MIME type of the page generated by the servlet that results from the JSP page

- **Notes**
  - Attribute value cannot be computed at request time
  - See section on response headers for table of the most common MIME types
Generating Excel Spreadsheets

<table>
<thead>
<tr>
<th>First</th>
<th>Last</th>
<th>Email Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marty</td>
<td>Hall</td>
<td><a href="mailto:hall@coreservlets.com">hall@coreservlets.com</a></td>
</tr>
<tr>
<td>Larry</td>
<td>Brown</td>
<td><a href="mailto:brown@coreservlets.com">brown@coreservlets.com</a></td>
</tr>
<tr>
<td>Steve</td>
<td>Balmer</td>
<td><a href="mailto:balmer@ibm.com">balmer@ibm.com</a></td>
</tr>
<tr>
<td>Scott</td>
<td>McNealy</td>
<td><a href="mailto:mcnealy@microsoft.com">mcnealy@microsoft.com</a></td>
</tr>
</tbody>
</table>

```<%@ page contentType="application/vnd.ms-excel" %>
<%-- There are tabs, not spaces, between cols. --%>
```

Conditionally Generating Excel Spreadsheets

• You cannot use the `contentType` attribute for this task, since you cannot make `contentType` be conditional.
  - The following *always* results in the Excel MIME type

```<% boolean usingExcel = checkUserRequest(request); %>
<% if (usingExcel) { %>
<%@ page contentType="application/vnd.ms-excel" %>
<% } %>
```

• Solution: use a regular JSP scriptlet with `response.setContentType`
Conditionally Generating Excel Spreadsheets (Code)

```html
...<BODY>
<CENTER>
<H2>Comparing Apples and Oranges</H2>
<% String format = request.getParameter("format");
if ((format != null) && (format.equals("excel"))) {
    response.setContentType("application/vnd.ms-excel");
}%
<TABLE BORDER=1>
  <TR><TH></TH>         <TH>Apples</TH> Oranges
  <TR><TH>First Quarter</TH> <TD>2307</TD> <TD>4706</TD></TR>
  <TR><TH>Second Quarter</TH> <TD>2982</TD> <TD>5104</TD></TR>
  <TR><TH>Third Quarter</TH> <TD>3011</TD> <TD>5220</TD></TR>
  <TR><TH>Fourth Quarter</TH> <TD>3055</TD> <TD>5287</TD></TR>
</TABLE>
</CENTER></BODY></HTML>
```

Conditionally Generating Excel Spreadsheets (Results)

![Excel spreadsheet comparing apples and oranges](image)
The session Attribute

• **Format**
  – `<%@ page session="true" %>` `<%-- Default --%>`
  – `<%@ page session="false" %>`

• **Purpose**
  – To designate that page not be part of a session

• **Notes**
  – By default, it is part of a session
  – Saves memory on server if you have a high-traffic site
  – *All* related pages have to do this for it to be useful

The isELIgnored Attribute

• **Format**
  – `<%@ page isELIgnored="false" %>`
  – `<%@ page isELIgnored="true" %>`

• **Purpose**
  – To control whether the JSP 2.0 Expression Language (EL) is ignored (true) or evaluated normally (false).

• **Notes**
  – If your web.xml specifies servlets 2.3 (corresponding to JSP 1.2) or earlier, the default is true
    • But it is still legal to change the default—you are permitted to use this attribute in a JSP-2.0-compliant server regardless of the web.xml version.
  – If your web.xml specifies servlets 2.4 (corresponding to JSP 2.0) or earlier, the default is false
The buffer Attribute

• Format
  – `<%@ page buffer="sizekb" %>`
  – `<%@ page buffer="none" %>`

• Purpose
  – To give the size of the buffer used by the `out` variable

• Notes
  – Buffering lets you set HTTP headers even after some page content has been generated (as long as buffer has not filled up or been explicitly flushed)
  – Servers are allowed to use a larger size than you ask for, but not a smaller size
  – Default is system-specific, but must be at least 8kb

The errorPage Attribute

• Format
  – `<%@ page errorPage="Relative URL" %>`

• Purpose
  – Specifies a JSP page that should process any exceptions thrown but not caught in the current page

• Notes
  – The exception thrown will be automatically available to the designated error page by means of the "exception" variable
  – The `web.xml` file lets you specify `application-wide` error pages that apply whenever certain exceptions or certain HTTP status codes result.
    • The errorPage attribute is for `page-specific` error pages
The isErrorPage Attribute

• Format
  – `<%@ page isErrorPage="true" %>`
  – `<%@ page isErrorPage="false" %>` -- Default --%

• Purpose
  – Indicates whether or not the current page can act as the error page for another JSP page

• Notes
  – A new predefined variable called exception is created and accessible from error pages
  – Use this for emergency backup only; explicitly handle as many exceptions as possible
    • Don't forget to always check query data for missing or malformed values

Error Pages: Example

```jsp
...<BODY>
<%@ page errorPage="/WEB-INF/SpeedErrors.jsp" %>
<TABLE BORDER=5 ALIGN="CENTER">
  <TR><TH CLASS="TITLE">Computing Speed</TABLE>
<%!
  private double toDouble(String value) {
    return(Double.parseDouble(value));
  }
%
  double furlongs = toDouble(request.getParameter("furlongs"));
  double fortnights =
    toDouble(request.getParameter("fortnights"));
  double speed = furlongs/fortnights;
%
<UL>
  <LI>Distance: <%= furlongs %> furlongs.
  <LI>Time: <%= fortnights %> fortnights.
  <LI>Speed: <%= speed %> furlongs per fortnight.
</UL>
</BODY></HTML>
```
Error Pages: Example (Continued)

...<BODY>
<%@ page isErrorPage="true" %>
<TABLE BORDER=5 ALIGN="CENTER">
   <TR><TH CLASS="TITLE">
      Error Computing Speed</TH></TR>
   <TR><TD>
      ComputeSpeed.jsp reported the following error: <I><%= exception %></I>. This problem occurred in the following place:
   </TD></TR>
   <TR><TD>
      <PRE>
      <%@ page import="java.io.*" %>
      <%= exception.printStackTrace(new PrintWriter(out)); %>
      </PRE>
      </TD></TR>
</TABLE>
</BODY></HTML>
The extends Attribute

• Format
  – `<%@ page extends="package.class" %>`

• Purpose
  – To specify parent class of servlet that will result from JSP page

• Notes
  – Use with extreme caution
  – Can prevent system from using high-performance custom superclasses
  – Typical purpose is to let you extend classes *that come from the server vendor* (e.g., to support personalization features), not to extend your own classes.

The isThreadSafe Attribute

• Format
  – `<%@ page isThreadSafe="true" %>` `<%-- Default --%>`
  – `<%@ page isThreadSafe="false" %>`

• Purpose
  – To tell the system when your code is not threadsafe, so that the system can prevent concurrent access
  • Normally tells the servlet to implement SingleThreadModel

• Notes
  – Avoid this like the plague
  • Causes degraded performance in some situations
  • Causes incorrect results in others
Example of Non-Threadsafe Code (IDs Must Be Unique)

- What's wrong with this code?

```jsp
<%! private int idNum = 0; %>
<% String userID = "userID" + idNum;
    out.println("Your ID is " + userID + ".");
    idNum = idNum + 1;
%>
```

Is isThreadSafe Needed Here?

- No! It is not needed. Synchronize normally:

```jsp
<%! private int idNum = 0; %>
<% synchronized(this) {
    String userID = "userID" + idNum;
    out.println("Your ID is " + userID + ".");
    idNum = idNum + 1;
}
%>
```

- Better performance in high-traffic environments
- isThreadSafe="false" will totally fail if server uses pool-of-instances approach
Summary

- **Used frequently**
  - `import`
    - Changes the packages imported by the servlet that results from the JSP page
      - *Always use packages for utility classes!*
  - `contentType`
    - Specifies MIME type of result
    - Cannot be used conditionally
      (use `<% response.setContentType(...) %>` for that)

- **Used moderately**
  - `isELIgnored`, `session`, `buffer`

- **Used occasionally**
  - `errorPage/isErrorpage`
  - `extends`

- **Avoid like the plague**
  - `isThreadSafe`
    - *Always use explicit synchronization instead*