

**Instructor Notes:**

Add instructor notes  
here.

# Java Server Pages (JSP)

Lesson 08 : JSP Actions



**Instructor Notes:**

This lesson explains  
the JSP Actions

### Lesson Objectives

In this lesson, you will learn the following concepts:

- JSP Actions
- jsp:include Action
  - include Action Vs Directive
- jsp:forward Action
  - Integrating Servlets & JSP
  - jsp:forward Vs response.sendRedirect



**Instructor Notes:**

Explain what is the functionality of JSP Action and how it is different from a directive.

Explain in brief the various actions available in JSP

8.1: JSP Actions

**The JSP Action**

JSP actions controls the behavior of the servlet engine.

Available Actions are as follows:

- `jsp:include`
- `jsp:forward`

**JSP Actions:****What is a JSP Action?**

- **JSP actions** use constructs in XML syntax to control the behavior of the servlet engine. With JSP actions, following actions can be performed:
  - Dynamically insert a file
  - Reuse JavaBeans components
  - Forward the user to another page
  - Generate HTML for the Java plugin
- Available actions include:
  - **jsp:include** : It includes a file at the time the page is requested.
  - **jsp:forward** : It forwards a client request to an HTML file, JSP file, or servlet for processing.
  - **jsp:useBean** : It finds or instantiates a JavaBean.
  - **jsp:setProperty** : It sets the property of a JavaBean.
  - **jsp:getProperty** : It inserts the property of a JavaBean into the output.
- The above actions are described in more detail further in this lesson.

## Instructor Notes:

Explain include action and its syntax. Mention that the attribute `flush="true"` is mandatory as it is not the default value. Also explain how parameters can be passed to `jsp:include` action using `<jsp:param>` sub element.

### 8.2: `jsp:include` Action The Include Action



The include action allows to insert files into the page being generated.  
Syntax:

```
<jsp:include page="{relativeURL | <%=  
expression%>}" flush="true"/>
```

or

```
<jsp:include page="{relativeURL | <%= expression  
%>}" flush="true" >  
<jsp:param name="parameterName" value="{parameterValue |  
<%= expression %>}" />+  
</jsp:include>
```

### The `jsp:include` Action:

- The **`<jsp:include>`** element allows to include either a **static** or **dynamic** file in a JSP file. The results of including static and dynamic files are quite different.
  - If the file is static, its content is included in the calling JSP file.
  - If the file is dynamic, it acts on a request and sends back a result that is included in the JSP page.
- When the include action is finished, the JSP container continues processing the remainder of the JSP file.
- We cannot always determine from a pathname if a file is static or dynamic. For example: `http://server:8080/index.html` might map to a dynamic servlet through a Web server alias. The **`<jsp:include>`** element handles both types of files, so it is convenient to use when we do not know whether the file is static or dynamic.
- If the included file is dynamic, we can use a **`<jsp:param>`** clause to pass the name and value of a parameter to the dynamic file. As an example, we can pass the string username and a user's name to a login form that is coded in a JSP file.
- The above slide lists the syntax of **`jsp:include`** action. The value for the page attribute is a **relative URL** that locates the file to be included, or an **expression** that evaluates to a String equivalent to the relative URL. The relative URL looks like a pathname - it cannot contain a protocol name, port number, or domain name. The URL can be absolute or relative to the current JSP file. If it is absolute (beginning with a `/`), then the pathname is resolved by Web or application server.
- We must include the attribute **`flush="true"`**, as it is not a default value. We cannot use a value of false. Use the **`flush`** attribute exactly as shown in the slide.

**Instructor Notes:**

Ask the participants to guess the difference in the URLs specified for the page attribute. Check with them whether they can recollect how a parameter passed to a servlet can be retrieved. In the similar fashion we can retrieve the parameter passed to a JSP page.

### 8.2: jsp:include Action Examples for jsp:include

**Example 1:**

```
<jsp:include page="scripts/login.jsp" />  
<jsp:include page="copyright.html" />  
<jsp:include page="/index.html" />
```

**Example 2:**

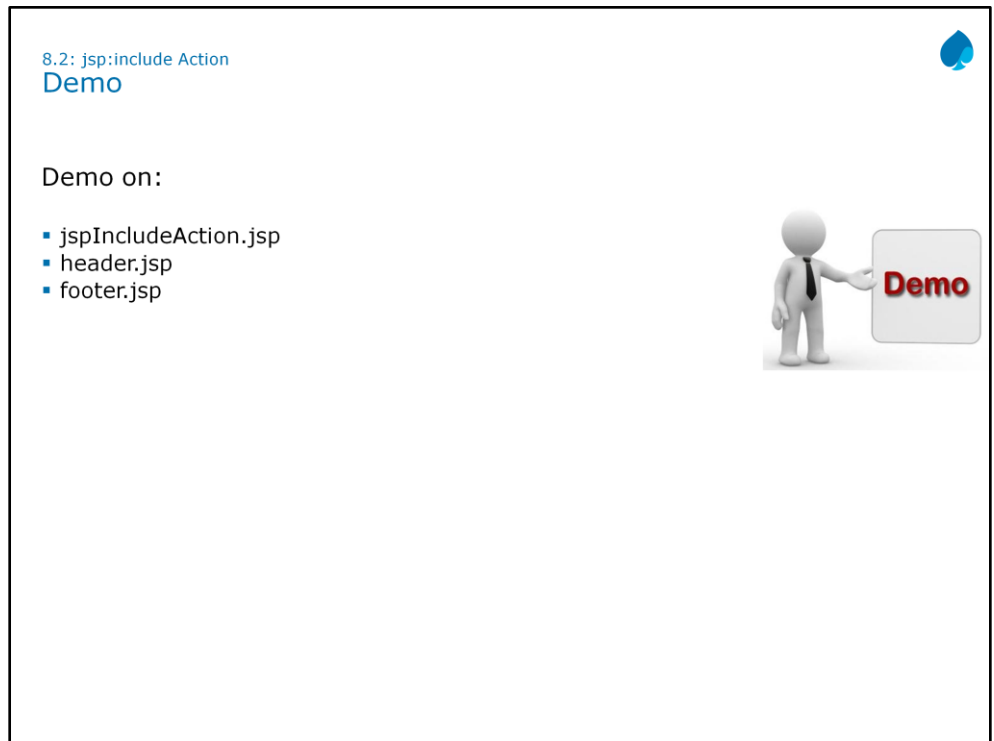
```
<jsp:include page="scripts/login.jsp">  
  <jsp:param name="username" value="jsmith" />  
</jsp:include>
```

## **The jsp:include Action: Examples for jsp:include:**

- The above slide lists some examples of **jsp:include** action with the page attribute value as a relative URL.
- In the second example on the slide, the **<jsp:param>** clause is used which allows to pass one or more name / value pairs as parameters to an included file. The included file should be dynamic, that is a JSP file, servlet, or other file that can process the parameter.
- We can use more than one **<jsp:param>** clause if we want to send more than one parameter to the included file. The name attribute specifies the parameter name and takes a case-sensitive literal string. The value attribute specifies the parameter value and takes either a case-sensitive literal string or an expression that is evaluated at request time.

**Instructor Notes:**

Demonstrate the examples for the `jsp:include` Action. Though the example is very simple it demonstrates clearly how a `jsp:include` action can be used.



Deploy web application **Lesson4-JSPActions** and show demo by executing each of the above JSP pages.

**Note:** Here is an example, where a JSP page uses `jsp:include` action to insert header and footer into `jspIncludeAction.jsp`

**Instructor Notes:**

Explain the difference between include Action & directive.  
Do mention that “A JSP container can include a mechanism for being notified if an included file changes, so the container can recompile the JSP page. However, the JSP 2.1 specification does not have a way of directing the JSP container that included files have changed”

## 8.2.1: include Action Vs Directive

**include Action versus include Directive**

The include directive includes files at the time the JSP page is translated into a servlet. Hence it is called static include.

- The include action includes files when the page is requested. Hence it is called dynamic include.
- In case of include directive as the include process is static, a change in the included file will not be reflected in the JSP file in which it is included (this behavior is dependent on JSP container).

**The jsp:include Action:****The JSP Include Action versus JSP Include Directive:**

- Unlike the **include directive**, which inserts the file at the time the JSP page is “translated” into a servlet, the **include action** inserts the file at the time the page is “requested”. This pays a small penalty in efficiency, and precludes the included page from containing general JSP code (for example, it cannot set HTTP headers), but it gains significantly in flexibility.
- A JSP container can include a mechanism for being notified if an included file changes (in case of include directive), so the container can recompile the JSP page. However, the JSP 2.2 specification does not have a way of directing the JSP container that included files that have changed.
- In the recent versions of the **Server containers** (such as JBOSS, Tomcat, Websphere Application server), this mechanism has been implemented. Hence there is no change in the behavior of **include directive** and **include action**. Both include the files dynamically, hence change in the included file is reflected in the original page in both the cases. However, it is recommended that we should not rely on the container’s behavior and always use **include action** whenever a **dynamic include** is required.

**Instructor Notes:**

Explain forward action. Do mention that unlike include the forward action would terminate the execution of the current page. The parameters can be passed to the forwarded page using `jsp:param` sub element. Do mention what will happen if we use `jsp:forward` action when the page output is not buffered. Forward is server-side forward. The difference between this action & `sendRedirect` is explained in the subsequent sections

8.3: `jsp:forward` Action**The forward Action**

The `<jsp:forward>` element forwards the request object containing the client request information from one JSP file to another file.

The target file can be an HTML file, another JSP file, or a servlet.

A `jsp:forward` effectively terminates the execution of the current page.

If the page output is buffered, then the buffer is cleared prior to forwarding.

**The `jsp:forward` Action:**

- The **`<jsp:forward>`** element forwards the **request object** containing the client request information from one JSP file to another file. The target file can be an HTML file, another JSP file, or a servlet, as long as it is in the same application context as the forwarding JSP file. The lines in the source JSP file after the **`<jsp:forward>`** element are not processed.
- We can pass parameter names and values to the target file by using a **`<jsp:param>`** clause. An example of this would be passing the parameter name ***username*** (with `name="username"`) and the value ***scott*** (with `value="scott"`) to a jsp login file as part of the request. If we use **`<jsp:param>`**, then the target file should be a dynamic file that can handle the parameters.
- Be careful while using **`<jsp:forward>`** with unbuffered output. If the **`<%@ page %>`** directive has used with **`buffer=none`** to specify that the output of the JSP file should not be buffered, and if the JSP file has any data in the out object, using **`<jsp:forward>`** will cause an `IllegalStateException`.




**Instructor Notes:**

Explain the syntax and examples listed on the slide.

8.3: jsp:forward Action

### Syntax and Examples for jsp:forward



Syntax:

```
<jsp: forward page="{ relativeURL | <%= expression %> }"
    {/}>|> [ <jsp:param name=" parameterName " value="{
    parameterValue | <%= expression %>}" /> ]+
</jsp: forward>
```

Example 1:

```
<jsp:forward page="/pages/login.jsp" />
```

Example 2:

```
<jsp:forward page="/pages/login.jsp">
    <jsp:param name="username" value="jsmith" />
</jsp:forward>
```

**The jsp:forward Action:**

The above slide lists the syntax for **jsp:forward** action:

- Like **jsp:include**, the value for the page attribute in case of jsp:forward is a **relative URL** or an **expression** representing the file to which we are forwarding the request. The file can be another JSP file, a servlet, or any other dynamic file that can handle a request object.
- The **jsp:param** element is similar to the one that we saw for jsp:include action.

**Examples:**

- The above slide shows examples of **jsp:forward** action.
  - The first example depicts how a request can be forwarded to another jsp page.
  - In the second example, the request is forwarded to another jsp and a request parameter "username" is also passed to the forwarded page. In the **login.jsp** page, the username request parameter can be obtained by using the following method:
    - request.getParameter("username")

**Instructor Notes:**

Demonstrate the examples for the forward Action and explain the code in brief. Invoke first.jsp page which will forward the request to second.jsp

### 8.3: jsp:forward Action Demo

Demo on:

- first.jsp
- second.jsp



## Deploy web application **Lesson4-JSPActions** and show demo by

```
<html>
<body bgcolor="white">
<font color="red"> VM Memory usage > 50%.
</html>
```

executing each of the above JSP pages in **forward** folder.

**Instructor Notes:**

Integrating Servlets and JSP means communication / information exchange from JSP to servlet and vice versa. From JSP to Servlet it can be done using forward action whereas from servlet to JSP it can be done using `RequestDispatcher.forward`. This integrating is a necessity for most of the web applications with MVC architectural pattern (JSP Model 2) where-in the view part taken care of by JSPs and the controller is implemented as a Servlet.

## 8.3.1: Integrating Servlets and JSP

**Integrating Servlets and JSP**

`jsp:forward` can be used to forward a request from jsp page to a servlet.

```
<!-- Forward to a servlet -->
<jsp:forward page="/servlet/servlettojsp" />
```

***Model 2 Architecture combines the use of Servlets and JSP. It takes advantage of the predominant strengths of both technologies, using JSP to generate the presentation Layer and servlets to perform process-intensive tasks.***

### **The `jsp:forward` Action:**

#### **Integrating Servlets and JSP:**


- The Model 2 architecture, discussed in lesson 1, is a hybrid approach for serving dynamic content, since it combines the use of both servlets and JSP. It takes advantage of the predominant strengths of both technologies, using JSP to generate the presentation layer and servlets to perform **process-intensive tasks**. On a more primitive level, we have seen how to integrate JSPs and servlets together using the **`jsp:forward`** tag.
- **Example:** A request to **`jsptoservlet.jsp`** forwards the request to the **`servletToJsp`** servlet which in turn sets the attribute in the request to contain the servlet name and obtains the **`RequestDispatcher`** to **`hello.jsp`** page and forwards the request using **`RequestDispatcher`**. Upon receiving the request, the **`hello.jsp`** shows the servlet name.
- The above slide shows the code fragment from the **`jsptoservlet.jsp`** file. The next slide shows the code listing for the **`servlet`** and **`hello.jsp`** files.

## Instructor Notes:

Demonstrate the integration between Servlets and JSP. Invoke `jsptoservlet.jsp` (<http://localhost:8080/jsp-demos/ch5/jsptoservlet>), which will forward the request to a servlet (i.e. `servletToJsp.java`) using forward action. The servlet then forwards the request to `hello.jsp` page. Thus, the response will be received from `hello.jsp` page as shown on the slide.

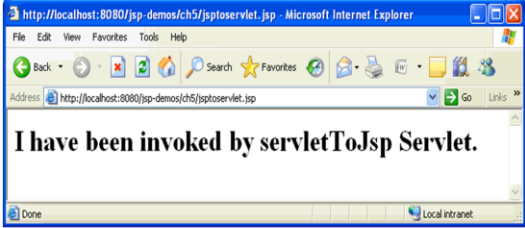

8.3.1: Integrating Servlets and JSP

### Demo: Integrating Servlets and JSP



Demo on:

- `servletToJsp.java` (servlet)

**Note:** Following is the code fragment from `servletToJsp` servlet which forwards the request and response to `hello.jsp` page.

```
public void doGet (HttpServletRequest request,
                  HttpServletResponse response) {
    try { // Set the attribute and Forward to hello.jsp
        request.setAttribute ("servletName", "servletToJsp");
        getServletConfig().getServletContext().getRequestDispatcher(
            "/pages/hello.jsp").forward(request, response);
    } catch (Exception ex) {
        ex.printStackTrace ();} }
```

The following example shows a code fragment from **hello.jsp**, which displays the servlet name passed by **servletToJsp** servlet.

```
<body bgcolor="white">
<h1> I have been invoked by
<% out.print (request.getAttribute("servletName").toString()); %>
Servlet. </h1>
```

## Instructor Notes:

Forward is server-side redirect whereas `sendRedirect` is client-side redirect. All the differences are listed on the slide and the notes page. Check with the participants what may be the application of these two (i.e. where to use which one ...). Use redirect to separate update components from display components. Suppose we have a `ProcessPaymentServlet` and a `ShowConfirmation.jsp` that informs the user that their payment has been processed. If we used a forward operation between them, if the user refreshed the page, *both* the servlet and the JSP would be re-executed, for a second payment.

On the other hand, if we used a client-side redirect, only the `ShowConfirmation.jsp` will be refreshed (which is probably what we want).

Forward operations are useful when we don't want the browser's address bar to be updated. Error pages are a good situation for using forward operations. Front Controllers in web frameworks that intercept all requests also use forward operations.

### 8.3.1: `jsp:forward` versus `response.sendRedirect`

#### `jsp:forward` versus `response.sendRedirect`



<code>jsp:forward</code>	<code>response.sendRedirect</code>
Server-side redirect, hence no network traffic	Client-side redirect, hence additional network round trip
The address of the destination page hidden from user	The address of the destination page visible to user in the address bar
Allows forwarding of the request to another page in the same context as the current page.	Allows re-direction of the request to another page in same or different context as the current page

### The `jsp:forward` Action:

#### `jsp:forward` versus `response.sendRedirect`:

- **forward** is server side redirect and **sendRedirect** is client side redirect. When we invoke a forward request, the request is sent to another resource on the server, without the client being informed that a different resource is going to process the request. This process occurs completely within the web container, and then returns to the calling method. When a **sendRedirect** method is invoked, it causes the web container to return to the browser indicating that a new URL should be requested. Since the browser issues a completely new request, any objects that are stored as **request attributes** before the redirect occurs will be lost. As a result of this extra round trip, a redirect is slower than forward. Client can disable `sendRedirect`.
- In case of **sendRedirect**, the user sees only the address of the destination page and can bookmark it and access it independently.
- In case of forward, the request can be forwarded to another page in the same context/domain as the current page whereas in case of **sendRedirect** we can redirect the request to a page in another context or domain as well. for e.g. if **sendRedirect** was called at `www.mydomain.com` then it can also be used to redirect a call to a resource on `www.yourdomain.com`.

**Instructor Notes:**

Lab on all the three forms of Scripting elements.

**Lab: Integrating JSP with Servlet****Lab 4.1**

**Instructor Notes:**

### Summary



In this lesson, you have learnt the following concepts:

- JSP Actions: include, forward, and plugin



**Instructor Notes:****Answers for the Review Questions:****Answer 1:** option 1**Answer 2:** flush=true**Answer 3:** jsp:param**Answer 4 :**  
jsp:forward**Review Question**

Question 1: The include action is a \_\_\_\_ include.

- Option 1: dynamic
- Option 2: static
- Option 3: both the above

Question 2: The \_\_\_\_ attribute of include action is mandatory to be set.

Question 3: \_\_\_\_ sub-element can be used to pass information to included or forwarded JSP page.

Question 4: \_\_\_\_ action terminates the execution of current page before moving on to the next page.

Question 5: \_\_\_\_ is a client-side redirect.

