JSP Action Elements

---

1. `<jsp:include>` - Includes a Resource

- **Description**: The `<jsp:include>` action is used to include a resource dynamically at **runtime**. This allows you to include other pages (like JSP, HTML, or even servlets) in the current JSP page. The content is inserted at the time of processing the request, which means the included resource can be updated independently of the JSP page itself.

  The included page could be another JSP or a static HTML file. This is particularly useful when you want to reuse content like a header, footer, or navigation menu across different pages.

Syntax:

<jsp:include page="relativeURL" />

-
- **Attributes**:
    - `page`: This attribute specifies the **URL** of the page that is to be included. The URL can be **relative** or **absolute**.

Example:

<jsp:include page="header.jsp" />
<h1>Welcome to My Web Page</h1>
<jsp:include page="footer.jsp" />

    -
        - In this example, `header.jsp` and `footer.jsp` are included before and after the main content respectively. This allows you to modularize the webpage and manage common elements separately.

---

2. `<jsp:forward>` - Forwards a Request

- **Description**: The `<jsp:forward>` action is used to **forward** the current request to another resource (a JSP, servlet, or HTML page). Unlike `<jsp:include>`, once the `forward` action is executed, the **current page processing stops**. This means the current page is not rendered anymore, and the response is completely handled by

the forwarded resource.

Forwarding is often used for:

- Redirecting users based on certain conditions (like login validation).
- Handling dynamic URLs where one page needs to pass control to another page.

Syntax:

```
<jsp:forward page="relativeURL" />
```

-
- **Attributes**:
    - `page`: Specifies the **URL** of the page to forward to (relative or absolute). Once forwarded, the request will be processed by the specified page.

Example:

```
<%
   if (someCondition) {
      request.setAttribute("message", "You have been forwarded!");
      forwardPage = "nextPage.jsp";
   }
%>
<jsp:forward page="<%= forwardPage %>" />
```

-
    - Here, based on a condition, the request is forwarded to `nextPage.jsp` where further processing happens. The `message` attribute set on the request object will also be available in the forwarded page.

---

3. `<jsp:useBean>` - Declares a JavaBean

- **Description**: The `<jsp:useBean>` action is used to **declare and instantiate a JavaBean** in a JSP page. A JavaBean is a reusable software component that follows specific conventions (e.g., having getters and setters). This action makes the bean accessible to the JSP page and stores it in one of the scopes (page, request, session, or application).

  If a bean already exists in the specified scope, no new bean is created; instead, the existing one is used. This is helpful when you want to access properties of a bean

directly from the JSP page.

Syntax:

<jsp:useBean id="beanName" class="beanClass" scope="page|request|session|application" />

-
- **Attributes**:
    - `id`: This is the name by which the bean is referenced in the JSP page.
    - `class`: This is the fully qualified class name of the JavaBean (e.g., `com.example.Student`).
    - `scope`: Defines the scope where the bean is stored. It can be `page`, `request`, `session`, or `application`. The default scope is `page`.

Example:

<jsp:useBean id="student" class="com.example.Student" scope="session" />
<jsp:setProperty name="student" property="name" value="John Doe" />
<jsp:getProperty name="student" property="name" />

-
    - In this example:
        - A `Student` bean is declared with the `id="student"`.
        - The `name` property of the `Student` bean is set to `"John Doe"`.
        - The `name` property is then retrieved and displayed on the page.

---

4. `<jsp:setProperty>` - Sets the Property of a Bean

- **Description**: The `<jsp:setProperty>` action is used to **set the value of a property** on a JavaBean that has already been declared using `<jsp:useBean>`. You can assign a value directly or bind it to a **request parameter** (if you want to set the property based on a form submission or URL parameter).

  This action is particularly useful when processing user input or dynamically setting values in JavaBeans.

Syntax:

```
<jsp:setProperty name="beanName" property="propertyName" value="value" />
```

-
- **Attributes**:
    - `name`: The name of the JavaBean.
    - `property`: The property of the JavaBean to be set.
    - `value`: The value to assign to the property (optional when binding to request parameters).

Example:

```
<jsp:useBean id="student" class="com.example.Student" />
<jsp:setProperty name="student" property="name" value="Jane Doe" />
```

-
    - The `name` property of the `student` bean is set to `"Jane Doe"`. This value can be retrieved and displayed later using `<jsp:getProperty>`.

---

5. `<jsp:getProperty>` - Gets the Property of a Bean

- **Description**: The `<jsp:getProperty>` action is used to **retrieve the value of a JavaBean's property** and display it directly on the JSP page. This is particularly useful for displaying the results of bean property values dynamically on the page.

Syntax:

```
<jsp:getProperty name="beanName" property="propertyName" />
```

-
- **Attributes**:
    - `name`: The name of the JavaBean from which to retrieve the property.
    - `property`: The specific property of the JavaBean to retrieve.

Example:

```
 <jsp:useBean id="student" class="com.example.Student" />
<jsp:setProperty name="student" property="name" value="Emily Smith" />
<p>Student Name: <jsp:getProperty name="student" property="name" /></p>
```

-
  - This will display the `name` property of the `student` bean (`"Emily Smith"`) on the page.

---

6. `<jsp:plugin>` - Embeds an Applet or a JavaBean

- **Description**: The `<jsp:plugin>` action is used to **embed applets or JavaBeans** into a JSP page. This action creates a plugin (usually a JavaBean or an applet) that can be embedded in the HTML output generated by the JSP.
  - **Applets**: Though not commonly used today due to browser security concerns, this action was originally used to embed Java applets.
  - **JavaBeans**: This is useful for embedding JavaBeans that interact with client-side components.

Syntax:

```
 <jsp:plugin type="bean|applet" code="beanClass" align="left|right|top|bottom">
   <jsp:param name="parameterName" value="parameterValue" />
</jsp:plugin>
```

-
- **Attributes**:
  - `type`: Specifies whether you are embedding a `bean` or an `applet`.
  - `code`: The class name of the applet or JavaBean to embed.
  - `align`: Specifies the alignment of the component (left, right, top, bottom).
  - `name`: The name of the parameter passed to the bean or applet.
  - `value`: The value of the parameter passed.

Example:

```
 <jsp:plugin type="bean" code="com.example.MyJavaBean" align="middle">
   <jsp:param name="width" value="500" />
```

```
    <jsp:param name="height" value="300" />
</jsp:plugin>
```

- This will embed the `MyJavaBean` class as a bean in the JSP page with the parameters `width` and `height`.

Summary of JSP Action Elements

Action Element	Purpose	Common Use
`<jsp:include>`	Includes another resource in the current page	Dynamic inclusion of content
`<jsp:forward>`	Forwards the request to another resource	Redirecting the request to another page
`<jsp:useBean>`	Declares a JavaBean and creates its instance	Declaring and creating JavaBeans
`<jsp:setProperty>`	Sets a property of a JavaBean	Setting properties of JavaBeans
`<jsp:getProperty>`	Retrieves a property value of a JavaBean	Displaying property values of JavaBeans
`<jsp:plugin>`	Embeds a JavaBean or applet into the page	Embedding JavaBeans or applets

---

Conclusion

JSP action elements are essential tools that help integrate dynamic content, forward requests, manage JavaBeans, and embed applets or JavaBeans into the web page. Understanding how to use these actions effectively will allow developers to build more modular, maintainable, and interactive JSP pages.