

# JSP

---

# Objectives

---

- ☐ Understand: What is JSP, Why JSP
- ☐ Explain: Life Cycle of JSP
- ☐ Introduction to JSP Tags
- ☐ Using Directives, Scripting Elements
- ☐ Working with Implicit Objects
- ☐ Understanding Standard Actions in JSP
- ☐ Introduction to Custom Tag, and The Need
- ☐ Creating Custom Tags
- ☐ Creating Custom Tags with Attribute

# What is JSP

---

- ❑ JSP stands for Java Server Pages.
- ❑ JSP is a server side component that is used to extend the functionality of web server.
- ❑ Used to generate dynamic web content.

# Why JSP

---

- ❑ Allows developers to concentrate on Presentation rather than Processing.
- ❑ Designers without knowing Java, still can develop elegant web pages.

# JSP Life Cycle

---

- There are 3 life cycle methods:
  - `jspInit()`
  - `_jspService()`
  - `jspDestroy()`

# JSP Tags

---

- JSP Specification supports 3 types of tags:
  - Directives
  - Scripting Elements
  - Standard Actions

# Directives

---

- Directives are divided into 3 categories:
  - page
  - include
  - taglib

# Page Directive

---

- ❑ Used to specify some information about the page.
- ❑ Syntax: `<%@page attr="value" ... %>`
- ❑ Important Attributes:
  - language
  - extends
  - import
  - session
  - buffer
  - autoflush
  - isThreadSafe
  - errorPage
  - isErrorPage
  - contentType



# Include Directive

---

- ❑ Used to include resources like HTML, JSP or Text files in JSP.
- ❑ Syntax:  
`<%@include file = "<filename>" %>`

# Scripting Elements

---

- Scripting Elements are divided into 3 categories:
  - Declaration
  - Scriptlet
  - Expression

# Declaration

---

☐ Used to declare variables and define methods.

☐ E.g.

```
<%!
```

```
    int x = 100;
```

```
    public void myMethod() {
```

```
        //Some Code
```

```
    }
```

```
%>
```

# Scriptlet

---

- ☐ Used to write any valid Java code.

- ☐ E.g.

```
<%
```

```
//Java Statements
```

```
%>
```

- ☐ Statements written inside scriptlet execute inside the service method of the servlet.

- ☐ Not possible to define methods inside scriptlet.

# Expressions

---

- ❑ Used to extract value of the variable.
- ❑ Expressions are directly processed on the browser window.
- ❑ E.g.  
`<%=<expr>%>`
- ❑ Methods returning 'void' cannot be invoked using expressions.

# Implicit Objects

---

- ☐ request – `javax.servlet.http.HttpServletRequest`
- ☐ response – `javax. servlet.http.HttpServletRequestResponse`
- ☐ out – `javax.servlet.jsp.JspWriter`
- ☐ session – `javax.servlet.http.HttpSession`
- ☐ config – `javax.servlet.ServletConfig`
- ☐ application – `javax.servlet.ServletContext`
- ☐ page – `java.lang.Object`
- ☐ pageContext – `javax.servlet.jsp.PageContext`
- ☐ exception – `java.lang.Throwable`

# Standard Actions

---

- ❑ Standard Actions are used to perform some specific task.
- ❑ All JSP standard actions follow a standard format:
  - `<prefix:suffix>`
- ❑ `suffix` is the actual name of the tag.

# <jsp:useBean>

---

- ❑ Used to instantiate a Java Bean.

- ❑ Important Attributes:

  - id

  - class

  - type

  - scope

- ❑ E.g.

```
<jsp:useBean id="d1"  
  class="java.util.Date"  
  scope="session" />
```



# <jsp:setProperty>

---

❑ Used to set properties of the bean.

❑ Syntax:

```
<jsp:setProperty name="<beanName>"  
.....<Property-Details>...../>
```

❑ Property Details:

■ `property=" * "`

■ `property="<property>"`

■ `property="<property>" param="<param>"`

■ `property="<property>" value="<value>"`

# <jsp:getProperty>

---

❑ Used to retrieve properties of the bean.

❑ Syntax:

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

# Other Actions

---

- ❑ `<jsp:forward>`
  - Used to forward the request to another page.
  - E.g. `<jsp:forward page="next.jsp" />`
- ❑ `<jsp:include>`
  - Used to include the resources in the existing JSP.
  - E.g. `<jsp:include page="next.jsp" />`
- ❑ `<jsp:param>`
  - Used in conjunction with either `<forward>` or `<include>`, to supply additional parameters.
  - E.g. `<jsp:param name="<name>" value="<value>" />`

# Custom Tags

---

- ❑ It's possible to create user defined tags.
- ❑ Need
  - `<jsp:useBean>` is capable of working upon only Value Objects.
  - To handle the processing, still developer is required to write a Java code inside scriptlets.
  - Custom tags can be used to reduce the no of scriptlets in JSP.

# Creating Custom Tags

---

- There are 2 resources required to create a custom tag:
  - Tag Handler Class
  - TLD (Tag Library Descriptor) File.

# Taglib Directive

---

❑ Used to locate TLD file in the JSP.

❑ E.g.

```
<%@taglib
```

```
    uri="<uri>"    prefix="<prefix>"
```

```
%>
```

# Creating Custom Tags with Attributes

---

- To create custom tags with attributes, 2 additional changes are required:
  - Add a property in the tag handler class that must match with the name of the attribute, with getters and setters.
  - Make the entry of attribute in TLD file.