(*) (*) UNIT-IV (JSP) (JSP Dir

JSP Directives: Page, Include & Taglib Tutorial

What are JSP Directives?

- JSP directives are the messages to JSP container. They provide global information about an entire JSP page.
- JSP directives are used to give special instruction to a container for translation of JSP to servlet code.
- In JSP life cycle phase, JSP has to be converted to a servlet which is the translation phase.
- They give instructions to the container on how to handle certain aspects of JSP processing
- Directives can have many attributes by comma separated as key-value pairs.
- In JSP, directive is described in <%@ %> tags.

Syntax of Directive:

```
<%@ directive attribute="" %>
```

There are three types of directives:

1. Page directive
2. Include directive
3. Taglib directive

JSP Page directive

Syntax of Page directive:

```
<%@ page...%>
```

- It provides attributes that get applied to entire JSP page.
- It defines page dependent attributes, such as scripting language, error page, and buffering requirements.
- It is used to provide instructions to a container that pertains to current JSP page.

Following are its list of attributes associated with page directive:

1. Language
2. Extends
3. Import
4. contentType
5. info
6. session
7. isThreadSafe
8. autoflush
9. buffer
10. IsErrorPage
11. pageEncoding
12. errorPage
13. isELIgonored

1. **language**: It defines the programming language (underlying language) being used in the page.

JSP Directives)

Syntax of language:

```
<%@ page language="value" %>
```

Here value is the programming language (underlying language)

Example:

```
<%@ page language="java" contentType="text/html%>
```

Explanation of code: In the above example, attribute language value is Java which is the underlying language in this case. Hence, the code in expression tags would be compiled using java compiler.

2. **Extends**: This attribute is used to extend (inherit) the class like JAVA does

Syntax of extends:

```
<%@ page extends="value" %>
```

Here the value represents class from which it has to be inherited.

Example:

```
<%@ page language="java" contentType="text/html %>

<%@ page extends="demotest.DemoClass" %>
```

Explanation of the code: In the above code JSP is extending DemoClass which is within demotest package, and it will extend all class features.

3. **Import**: This attribute is most used attribute in page directive attributes.It is used to tell the container to import other java classes, interfaces, enums, etc. while generating servlet code.It is similar to import statements in java classes, interfaces.

Syntax of import:

```
<%@ page import="value" %>
```

Here value indicates the classes which have to be imported.

Example:

```
<%@ page language="java" contentType="text/html%>
```

Explanation of the code:

In the above code, we are importing Date class from java.util package (all utility classes), and it can use all methods of the following class.

4. **contentType**:

- It defines the character encoding scheme i.e. it is used to set the content type and the character set of the response
- The default type of contentType is "text/html; charset=ISO-8859-1".

Syntax of the contentType:

```
<%@ page contentType="value" %>
```

Example:

```
<%@ page language="java" contentType="text/html%>
```

Explanation of the code:

In the above code, the content type is set as text/html, it sets character encoding for JSP and for generated response page.

5. **info**

- It defines a string which can be accessed by getServletInfo() method.
- This attribute is used to set the servlet description.

Syntax of info:

```
<%@ page info="value" %>
```

Here, the value represents the servlet information.

Example:

```
<%@ page language="java" contentType="text/html; info="Sreyas Directive JSP"
pageEncoding="ISO-8859-1"%>
```

Explanation of the code:

In the above code, string "Sreyas Directive JSP" can be retrieved by the servlet interface using getServletInfo()

6. **Session**

- JSP page creates session by default.
- Sometimes we don't need a session to be created in JSP, and hence, we can set this attribute to false in that case.The default value of the session attribute is true, and the session is created.

  When it is set to false, then we can indicate the compiler to not create the session by default.

Syntax of session:

```
<%@ page session="true/false"%>
```

Here in this case session attribute can be set to true or false

Example:

```
<%@ page language="java" contentType="text/html;session="false"%>
```

Explanation of code:

In the above example, session attribute is set to "false" hence we are indicating that we don't want to create any session in this JSP

7. **isThreadSafe:**

- It defines the threading model for the generated servlet.
- It indicates the level of thread safety implemented in the page.
- Its default value is true so simultaneous
- We can use this attribute to implement SingleThreadModel interface in generated servlet.
- If we set it to false, then it will implement SingleThreadModel and can access any shared objects and can yield inconsistency.

Syntax of isThreadSafe:

```
<% @ page isThreadSafe="true/false" %>
```

Here true or false represents if synchronization is there then set as true and set it as false.

Example:

```
<%@ page language="java" contentType="text/html;%>
```

Explanation of the code:

In the above code, isThreadSafe is set to "true" hence synchronization will be done, and multiple threads can be used.

8. **AutoFlush:**

This attribute specifies that the buffered output should be flushed automatically or not and default value of that attribute is true.

If the value is set to false the buffer will not be flushed automatically and if its full, we will get an exception.

When the buffer is none then the false is illegitimate, and there is no buffering, so it will be flushed automatically.

Syntax of autoFlush:

```
<% @ page autoFlush="true/false" %>
```

Here true/false represents whether buffering has to be done or not

Example:

```
<%@ page language="java" contentType="text/html;autoFlush="false"%>
```

Explanation of the code:

In the above code, the autoflush is set to false and hence buffering won't be done and it has manually flush the output.

9. **Buffer:**

- Using this attribute the output response object may be buffered.
- We can define the size of buffering to be done using this attribute and default size is 8KB.
- It directs the servlet to write the buffer before writing to the response object.

Syntax of buffer:

```
<%@ page buffer="value" %>
```

Here the value represents the size of the buffer which has to be defined. If there is no buffer, then we can write as none, and if we don't mention any value then the default is 8KB

Example:

```
<%@ page language="java" contentType="text/html; buffer="16KB"%>
```

Explanation of the code:

In the above code, buffer size is mentioned as 16KB wherein the buffer would be of that size

10. **isErrorPage:**

- It indicates that JSP Page that has an errorPage will be checked in another JSP page
- Any JSP file declared with "isErrorPage" attribute is then capable to receive exceptions from other JSP pages which have error pages.
- Exceptions are available to these pages only.
- The default value is false.

Syntax of isErrorPage:

```
<%@ page isErrorPage="true/false"%>
```

Example:

```
<%@ page language="java" contentType="text/html; isErrorPage="true"%>
```

Explanation of the code:

In the above code, isErrorPage is set as true. Hence, it will check any other JSPs has errorPage (described in the next attribute) attribute set and it can handle exceptions.

11. **PageEncoding:**

The "pageEncoding" attribute defines the character encoding for JSP page.

The default is specified as "ISO-8859-1" if any other is not specified.

Syntax of pageEncoding:

```
<%@ page pageEncoding="vaue" %>
```

Here value specifies the charset value for JSP

Example:

```
<%@ page language="java" contentType="text/html;" pageEncoding="ISO-8859-1"
    isErrorPage="true"%>
```

Explanation of the code:

In the above code "pageEncoding" has been set to default charset ISO-8859-1

12. **errorPage:**

This attribute is used to set the error page for the JSP page if JSP throws an exception and then it redirects to the exception page.

Syntax of errorPage:

```
<%@ page errorPage="value" %>
```

Here value represents the error JSP page value

Example:

```
<%@ page language="java" contentType="text/html;" pageEncoding="ISO-8859-1"
    errorPage="errorHandler.jsp"%>
```

Explanation of the code:

In the above code, to handle exceptions we have errroHandler.jsp

13. **isELIgnored:**

- IsELIgnored is a flag attribute where we have to decide whether to ignore EL tags or not.
- Its datatype is java enum, and the default value is false hence EL is enabled by default.

Syntax of isELIgnored:

```
<%@ page isELIgnored="true/false" %>
```

Here, true/false represents the value of EL whether it should be ignored or not.

Example:

```
<%@ page language="java" contentType="text/html;" pageEncoding="ISO-8859-1"
    isELIgnored="true"%>
```

Explanation of the code:

In the above code, isELIgnored is true and hence Expression Language (EL) is ignored here.

In the below example we are using four attributes(code line 1-2)

JSP Include directive

- JSP "include directive" is used to include one file to the another file
- This included file can be HTML, JSP, text files, etc.
- It is also useful in creating templates with the user views and break the pages into header&footer and sidebar actions.
- It includes file during translation phase

Syntax of include directive:

```
<%@ include….%>
```

Example:

Directive_jsp2.jsp (Main file)

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3.     <%@ include file="directive_header_jsp3.jsp" %>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Guru Directive JSP2</title>
9. </head>
10.    <body>
11.    <a>This is the main file</a>
12.    </body>
13.    </html>
```

Directive_header_jsp3.jsp (which is included in the main file)

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <html>
4. <head>
5. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
6.
7. </head>
```

```
8. <body>
9. <a>Header file : </a>
10.     <%int count =1; count++;
11.     out.println(count);%> :
12.     </body>
13.     </html>
```

JSP Taglib Directive

- JSP taglib directive is used to define the tag library with "taglib" as the prefix, which we can use in JSP.
- More detail will be covered in JSP Custom Tags section
- JSP taglib directive is used in the JSP pages using the JSP standard tag libraries
- It uses a set of custom tags, identifies the location of the library and provides means of identifying custom tags in JSP page.

Syntax of taglib directive:

```
<%@ taglib uri="uri" prefix="value"%>
```

Here "uri" attribute is a unique identifier in tag library descriptor and "prefix" attribute is a tag name.

Example:

```
1. <%@ page language="java" contentType="text/html; %>
2.     <%@ taglib prefix="sreyastag" uri="http://java.sun.com/jsp/jstl/core" %>
3. <html>
4. <head>
5. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
6. <title>Guru Directive JSP</title>
7. <sreyastag:hello/>
8. </head>
9. <body>
10.     </body>
11.     </html>
```

Explanation of the code:

Code Line 3: Here "taglib" is defined with attributes uri and prefix.

Code Line 9: "sreyastag" is the custom tag defined and it can be used anywhere