UNIT - 4

JSP (Java Server page)

1. Introduction to JSP ? Advantages of JSP over Servlet ? Architecture (or) processing JSP, lifecycle of JSP ? Simple JSP Code Writing and Executing ?

2. Cookies and Sessions in JSP ?

3. Connecting to DATABASE in JSP ?

4. Using Beans in JSP ?

5. Anatomy (or) Components of JSP ? [Scriptlet tag, Expression tag, Declaration tag, Action tag, Custom tag Directives]

6. Directives [page directives, include directives, taglib directive]

7. Implicit objects.

① Introduction to JSP

↓

Java Server pages

⇒ Java Server pages (JSP) is a technology that helps Software developers to Create dynamic webpages based on HTML, XML, (or) other document types.

⇒ Using JSP, one can easily Separate presentation logic and business logic. and java developers Can write Server side Complex Computation Code without Concerning the web design.

⇒ JSP's are released in the year 1999 by Sun micro Systems JSP is Similar to the Java programming language. To deploy and run java Server pages, a Compatible web Server with a Servlet Container Such as Apache tomcat is required

$\boxed{\text{JSP = Html + java Code embedded}}$

Advantages of JSP over Servlet:

⇒ JSP technology is used to Create web Application just like Servlet technology. JSP is an extension to Servlet because it provides more functionalities than Servlet. Such as expression language, JSTL etc...

A JSP page Consists of HTML tags and JSP tags. the JSP pages are easier to maintain than Servlet because we can Separate designing and development. It provides Some additional features Such as Expression language, Custom Tags etc.

There are many advantages of JSP over the Servlet. they are as follows.

① Extension to Servlet

JSP technology is the extension to Servlet technology. we Can use all the features of the Servlet in JSP. in addition to, we Can use implicit objects, predefined tags, expression language, Custom tags in JSP, that makes JSP development easy.

② Easy to maintain:

JSP can be easier to manage because we Can easily Separate our business logic with presentation logic, In Servlet technology, we mix our business logic with presentation logic.

③ Fast Development :- No need to recompile and redeploy

If JSP page is modified, we dont need to recompile and redeploy the project. the Servlet Code needs to be updated and recompiled if we have to change the look and feel of an application

④ Less Code than Servlet:

In JSP we Can use many tags Such as action tags, JSTL, Custom-tags etc that reduces the Code.

Life Cycle of a JSP page & Architecture of JSP (or) JSP processing

The JSP pages follow these phases :

the web Server needs a JSP Engine or JSP Container to process JSP pages. the JSP Container is Responsible for taking request of JSP page.

1. Translation of JSP page
2. Compilation of JSP page
3. class loading [the class loader loads class file]
4. Instantiation [object of Generated Servlet is Created]
5. initialization [the Container calls jsp init() method]
6. Request processing [the Container invokes jsp Service() method] $\underset{\text{calls}}{\downarrow}$
7. Destroy [the Container invokes jsp Destroy() method] $\underset{\text{calls}}{\downarrow}$

life cycle methods

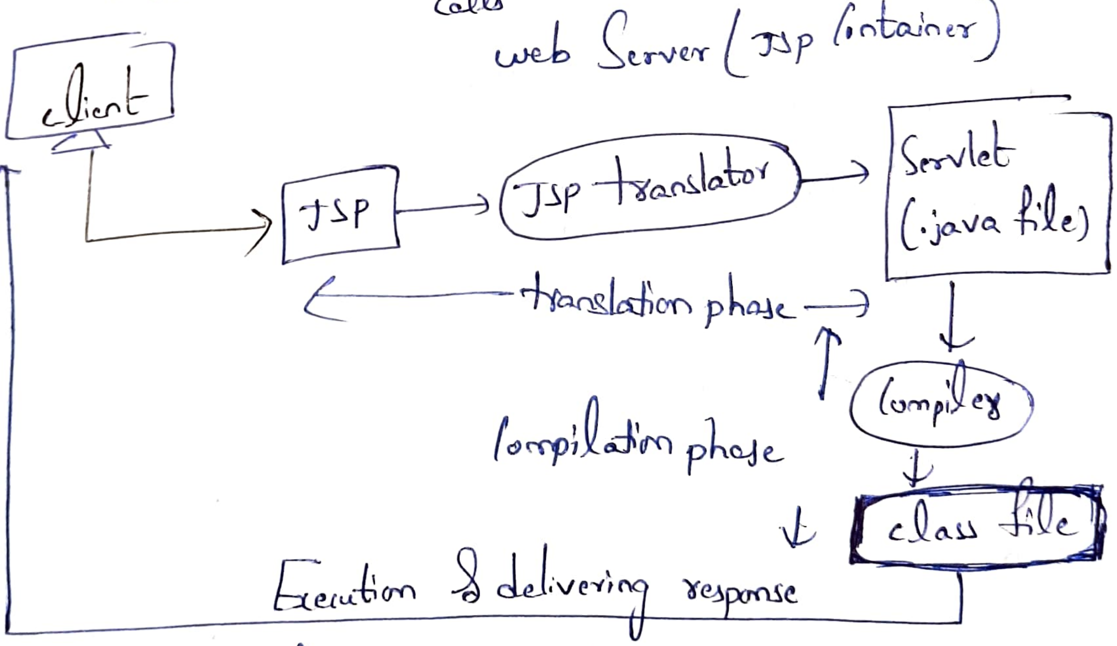


fig: JSP Architecture

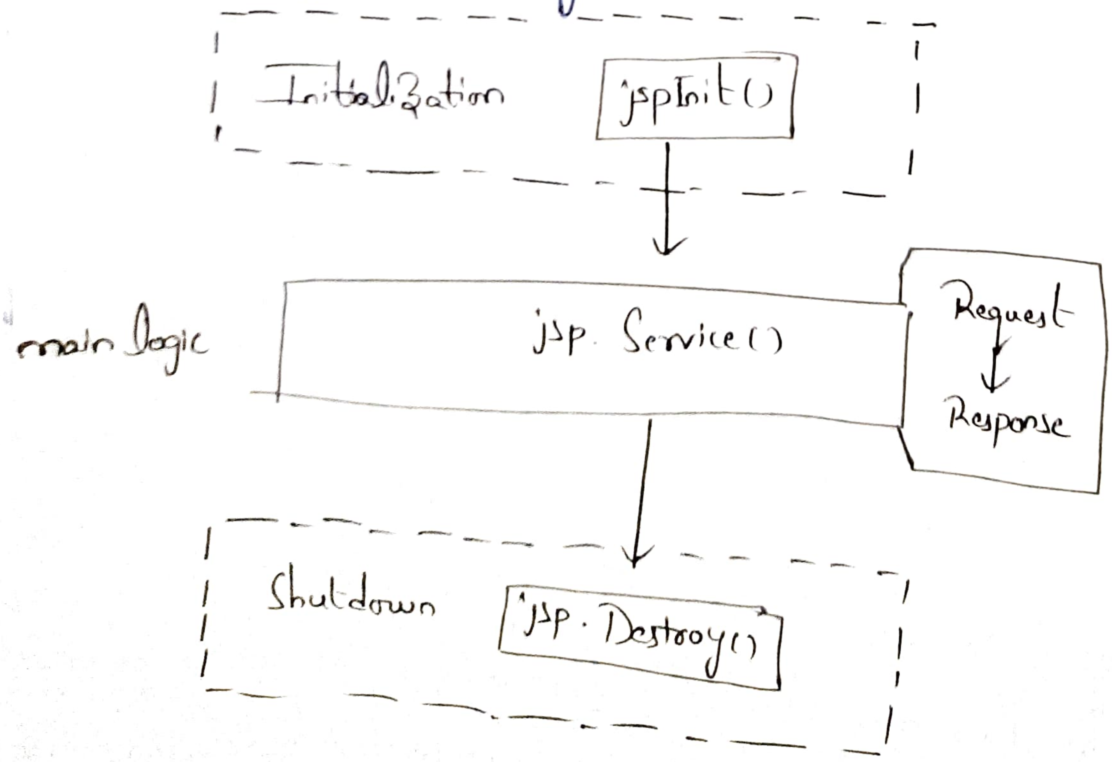
As shown in the above diagram, JSP page is translated into Servlet by the help of JSP translator. the JSP translator is a part of web Server which is responsible for translating the JSP page into Servlet. After that, Servlet page is Compiled by the Compiler and gets Converted into the class file. the class file is executed and output of execution is Sent to client as response.

Now lets See the life Cycle of Java Server page (JSP)

<u>JSP life Cycle:-</u>

A JSP life Cycle Can be defined as the entire process from its Creation to till its destruction which is Similar to the Servlet life Cycle with a additional Step which is required to Compile a JSP into Servlet.

the major phases of JSP life Cycle are Very Similar to Servlet life Cycle and they are as follows.



① JSP Compilation:

when a browser asks for a JSP, the JSP engine first check. to See whether it needs to Compile the page. If the page has ~~to~~ never been Compiled. or if the jsp has been modified Since it was last Compiled. the JSP engine Compiles the page.

② JSP. Initialization :-

When a Container loads a JSP it invokes the jsp Init() method
calls
before Servicing any requests. If you need to perform JSP- Specific initialization use jspInit() method

```
public Void jspInit ()
   {
     // initialization Code ----

   }
```

Typically initialization is performed only once ~~and as with~~

③ JSP Execution:-

JSP Service() method is used to Serve the raised requests by JSP. It takes request and response objects as parameters
ie, It takes HttpServlet Request and HttpServlet Response as its parameters.

The JSP Service () method is ~~Called~~ invoked once per each request and is responsible for generating one response for the request.

```
~~JSP~~  Void  JspService (HttpServletRequest request,
                           HttpServletResponse response)
         {
           // Service handling code
         }
```

④ JSP cleanup

⇒ The destroy phase of JSP life cycle ~~represents~~ is used when we want to remove JSP from Container. This method is called only once, if you need to perform cleanup tasks like closing file releasing database Connection etc..

⇒ jsp Destroy () method is used inorder to destroy the method

```
public Void  jsp Destroy ()
   {
     // your cleanup Code goes here....
   }
```

SIMPLE JSP CODE WRITING AND EXECUTING

for executing JSP code we must have

① JDK installed

② Apache tomcat installed

Step 1: open text editor and type following Code hello.jsp

```
<html>
<body>
<% out.println ("this is my first JSP page! "); %>
</body>
</html>
```

$\boxed{\text{o/p:- this is my first JSP page}}$

Create a Separate directory at the path c:\your-tomcat-directory\webapps\jsp-examples & store above code in newly created directory.

I have created a directory named HelloDemo in which I have stored above program by name hello.jsp. Note that while Saving file using Notepad editor with .jsp extension, you must Select all files option. If you donot do that then file may saved as txt file because the default extension for notepad is txt.

Step-2:-

start Tomcat web Server by typing Command Startup at Command prompt or by clicking Startup file.

Step-3:

Open Some ~~web~~ web browser like firefox or IE. Type path for JSP page prefix http://localhost:8080. Note that localhost is default DNS for tomcat webServer.

http://localhost:8080/foldername/yourpgm.extension i.e

http://localhost:8080/examples/hello.jsp.

2. Using Cookies and Session for Session tracking

JSP Cookies handling:

Cookies are text files Stored on client Computer and they are kept for Various information tracking purposes. JSP Supports HTTP Cookies. using Servlet technology.

there are three steps Involved in identifying and returning users.

① Servlet Script Sends a Set of Cookies to browser. For example, name, age, Idnumber etc.

② Browser Stores this information on local machine for future use.

③ when the next time the browser Sends any request to the web Server then it Sends those Cookies information to Server and Server uses that information to identify user or may be for Some other purpose as well.

Setting Cookies with JSP'.

Setting Cookies with JSP involves three steps

ex Step(1): Creating a Cookie object

you call the Cookie Constructor with a Cookie name and a Cookie value, both of which are Strings. Keep in mind, neither the name nor the value should Contains white space or any of the characters like [ ], ( ), =, ,, /, ?, @, :, ;.

```
Cookie cookie = new Cookie("Key", "value");
```

Step②: Setting the maximum age

you use SetMaxAge to Specify how long (in Seconds) the Cookie Should be valid. the following Code will Setup a Cookie for 24 hours.

```
cookie.SetMaxAge(60*60*24);
```

Step③: Sending the Cookie into the HTTP response header

you use response.addCookie to add Cookies in the HTTP response header as follows.

```
response.addCookie(cookie);
```

Types of Cookies in JSP

2types

persistent Cookies
(or)
Permanent Cookies

they remain on hard drive and present until the user delete them or they expire themselves

Session Cookies
(or)
temporary Cookie

They get deleted thems as soon as the Session ends or browser close

Structure of Cookies in JSP :

A Cookies sent by a JSP page in HTTP header looks like thi

HTTP/1.1 200 OK

Date : Sat, 25 Nov 2021 10:03:38 GMT

Server : Apache tomcat /9.0.34 (windows)

Set-cookie : name = my-name ; expires = Sun, 26-Nov 2021, 10:03:38 GM

path = // ; domain = Sai.com

Connection : close

Content-Type = text / html.

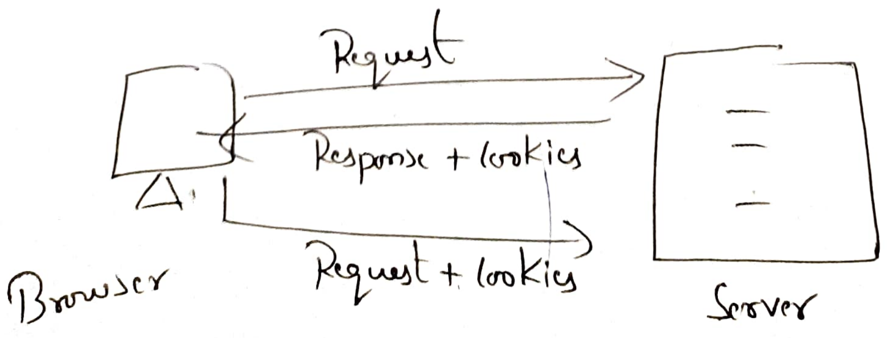
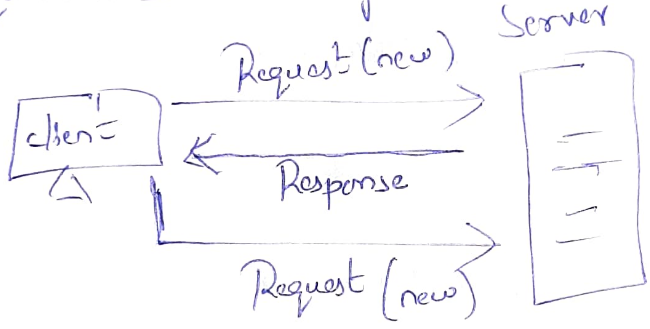


fig: Structure of Cookies.

Sessions in JSP :-

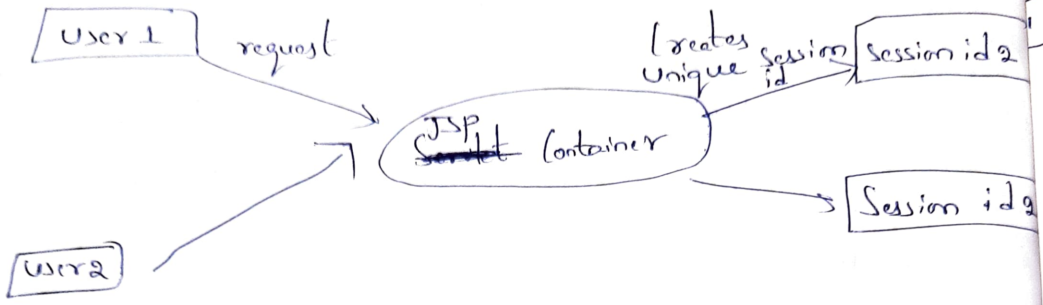
Session Simply means a particular interval of time. Session tracking is a way to mantain State(data) of an user. It is also Known as Session management in JSP. Http protocol is a stateless So we need to maintain State using Session tracking technique. Each time user requests to the Server. Server treats the request as the new request, So we need to maintain the State of an user to recognize to particular user.

http is a stateless that means each request is Considered as the new request is shown in fig.



Inorder to Achieve Session tracking in JSP, Cookies have been one of the most Commonly used technique however they have the following disadvantage

1. They Can Keep only textual information
2. they are browser dependent. Hence, if client disable them, your web application Cant make use of them
3. Individual Cookies Can Contain not more than 4kb of information.



Advantages of http Sessions in ~~Servlets~~ JSP

1. Any Kinds of object can be stored into a Session, be it a text, database, dataset etc.
2. Usage of Sessions is not dependent on clients browser
3. Sessions are Secure and transparent.

Disadvantages of http Session:

1. performance overhead due to Session object being stored on Server.
2. overhead due to Serialization and deserialization of data

there are four techniques used in Session tracking.

1. Cookies
2. hidden Form field
3. URL Rewriting
4. http Session object.

1) **Cookies**:-

web Server Can assign a unique Session ID as a Cookie to each client and for Sub Sequent requests ~~for~~ from the client they Can be recognized using the received Cookies.

2) **Hidden Form Fields**:

A web Server Can Send a hidden HTML form field along with a Unique Session ID as follows

```
<input type = " hidden name = "Sessionid" value = "12345" >
```

the entire meaning is that, when the form is Submitted, the Specific name and value are automatically included in the GET and POST data. Each time when web browser Sends requests back, then Session id value, Can be used to keep the track of different web browser.

3) **URL Rewriting**:-

you Can add Some extra data on the end of URL that identifies the Session.

Example: https:// www. . . . . .Com/ file.htm ; Sessionid=123

Sessionid to identify user

(4) Http Session object:

Servlet provides http Session interface which provides a way to identify a user across more than one page request or visit to a web Site and to store information about user

Connecting to DATABASE in JSP :-

The database is used for storing various types of data which are huge and has storing capacity in giga bytes. JSP can connect with such databases to create and manage the records.

while accessing JSP database ~~wes~~ from JSP page we should have some DB packages installed. In this section we will discuss the connectivity of JSP with my SQL data base.

Prerequirements :-

① Tomcat web Server

② My SQL Server

③ JDK

Step ① :- Creating a database named Students in MySQL using following commands.

```
mysql > CREATE DATABASE Students ;
```

then create a table named Students_table in the Students database as follows:

```
mysql > use Students ;

mysql > CREATE TABLE Students_table (
            roll_no INT(4) NOT NULL
                AUTO_INCREMENT )
```

name VARCHAR(50) NOT NULL,
address VARCHAR(50) NOT NULL,
phoneno VARCHAR(15) NOT NULL,
PRIMARY KEY (roll_no)

~~Step~~②: Student database: Students table

name	address	phoneno	rollno

Step②: for establishing the Connectivity between JSP & MySQ
using JDBC driver. what we need is to download [MySQL JDBC
Connector]. Download this Connector from [http://www.mysql.com/mysql
products/Connector/j/]. Just pickup the mirror and download
ZIP file. then from the extracted folder just copy jar file
name [mySql - Connector - java - xx - bin . jar] to
[C:/your-tomcat-directory/common/lib]. then just Set CLASS
using environmental Variables for that purpose goto Control pannel→Syst
System properties → environmental Variables & Set classpath

Variable name : [CLASSPATH]

Variable value : c:/your-directory/common/lib/mysql-Connector-java-3.1.19-bin

[OK] [Cancel]

Using Beans in JSP :-

Java beans are reusable Components. we Can use Simple java bean in JSP. this helps us in Keeping the business logic Seperate from presentation logic. Beans are used in JSP as instance of class. we must Specify Scope of the Bean in JSP page. Here Scope of bean means. how much time bean exists in JSP. when the bean is present in Scope its id is available in that Scope.

there are Various Scopes using which the bean Can be used in JSP. ~~page~~ They are.

1) page Scope :- It is default Scope. The bean object gets disappear as Soon as Current page gets ~~dis~~ Closed.

2) Request Scope :- the bean object remains in existence as long as the request object is present

3) Session Scope :- Bean object remains from Starting to ending time of user in internet on Browser.

4) Application Scope :- It ~~b~~ is broadcast Scope provided by JSP. It should be used only once when it is necessary.

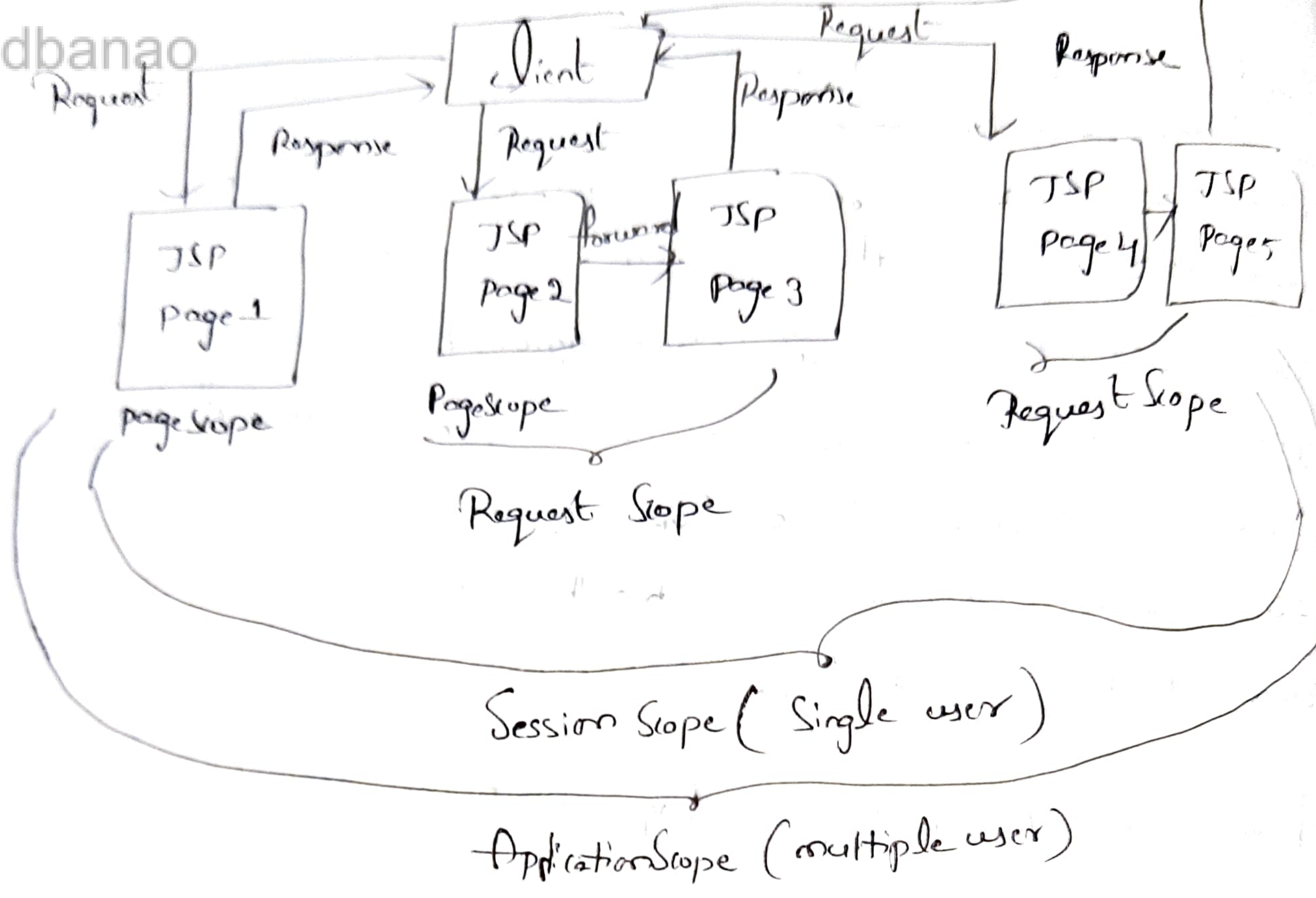


fig: Various Scopes.

Components of JSP Bean :-

① <jsp:usebean> It is nothing but a reusable software component
It is used to specify scope of bean.

② <jsp: Set property> this action tag is used to Set the values to bean class ~~by calling~~ properties by calling Setter methods.

<jsp: Set property> tag must be used inside of <jsp:useBean> tag

③ <jsp :getproperty>
this tag is used to read the values of a bean class property by calling getter method.
this tag must be used out Side of <jsp: UseBean> tag.

Anatomy of JSP / Components of JSP :-

JSP is built using Components Such as

1. Scriptlets tag
2. Expressions tag
3. Declarations tag

} Scripting element → provides ability to insert java code inside the jsp.

4. Action tags
5. Custom tags
6. Directives

1. Scriptlet tag :-

⇒ In JSP, java Code can be written inside the JSP page using the Scriptlet tag. A scriptlet tag is used to execute java Source Code in JSP.

```
Syntax :-  <% java Source Code %>
```

Example of JSP Scriptlet tag :-

```
<html>
<body>
<% out.print ("JSP is equal to HTML + Java"); %>
</body>
</html>
```

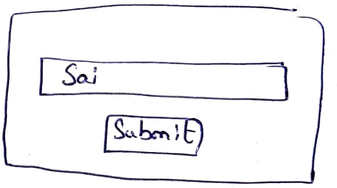
Example of JSP Scriptlet tag that prints the user name

in this example we have Created two files
1. index.html
2. welcome.jsp

The index.html file gets the username from the user and the welcome.jsp file prints the username with welcome message

index.html

```
<html>
<body>
<form  action = "welcome.jsp">
<input type = "text" name =" username">
<input type = "Submit" value = "go" > <br/>
</form>
</body>
</html>
```



welcome.jsp

```
<html>
<body>
<%
String name = request.getParameter("username");
out.print("welome" + name);
%>
</body>
</html>
```

o/p: welcome Sai

② Expression tag:

the Code placed within JSP expression tag is written to generate output as response. So you need ~~to write~~ out not to write out.print() to write data. it is mainly used to print the values of Variable or method

$\boxed{\text{Syntax :- <\% = Statement \%>}}$

Example:

in this example of jsp expression tag, we are Simply displaying a welcome message

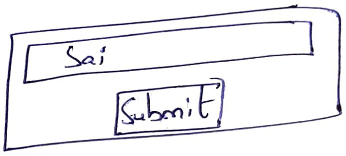
```
<html>
  <body>
    <% = " welcome to jsp" %>
  </body>
  </html>
```

Example of jsp expression tag that prints the user name

In this example, we are printing the username using the expression tag. the index.html file gets the user name and Sends the request to welcome.jsp file. which display user name.

index. html

```
<html>
<body>
<form   action  =  "welcome. jsp">
<input  type  =  "text"   name  =  "username">
<input  type  =  "Submit"   value  =  "go">  <br>
</form>
</body>
</html>
```



welcome.jsp:

```
<html>
<body>
<%= "welcome" + request.getparameter("username")%>
</body>
</html>
```

o/p: welcome Sai

③ Declaration tag:-

⇒ The JSP Declaration tag is used to declare fields and methods.

⇒ The Code written inside the jsp declaration tag is placed outside the Service() method of Servlet So it doesn't occupy memory for each request.

Syntax: <%! field or method declaration %>

Difference between JSP Scriptlet tag and Declaration tag

JSP Scriptlet tag	JSP Declaration tag
it can declare only Variables not methods	it can declare Variables as well as methods
the declaration, is placed inside jspService() method	the declaration is placed outside the Service method

Example:

```
<html>
<body>
<%! int data = 50; %>
<%= "value of Variable is " + data %>
</body>
</html>
```

o/p: Value of Variable is 50

JSP Directives:

The jsp directives are messages that tells the web container how to translate a jsp page into the corresponding Servlet.

There are three types of directives.

1. page directives
2. include directive
3. taglib directive

Syntax of JSP directive : <%@ directive attribute ="value"%>

1. JSP page directive:

The page directive defines attributes that are applied to an entire jsp page.

Syntax: <%@ page attribute ="value" %>

Attributes in JSP page directives are:

1. import
2. Content type
3. extends
4. info
5. buffer
6. language
7. is ELIgnored
8. ~~at~~ auto Flush
9. Session
10. page Encoding
11. error page
12. is Error page

① import:-

The import attribute is used to import class, interface or all the packages. It is similar to import keyword in java class or interface

```
Example: <% @ page import = "java.util.Data" %>
```

② Content type:-

The ContentType attribute defines the MIME (multipurpose internet Mail Extension) type of the HTTP response.

```
Example: <% @page Contenttype = application/msword %>
```

3) extends:-

The extends attributes defines the parent class that will be inherited by Servlet. It is rarely used

1) info:-

This attribute simply sets the information of JSP page.

```
Example:- <% @ page info = "Composed by Nagendra" %>
```

5 buffer:-

The buffer attribute sets the buffer size in kilo bytes to handle output generated by JSP page. The default size of buffer is 8 Kb.

```
Example: <% @ page buffer = "16 kb" %>
```

⑥ language:

the language attribute Specify the Scripting language used in JSP page. default value is "Java".

⑦ is ELIgnored:

we can ignore the Expression Language (EL) in jsp by the is ELIgnored Attribute.

⑧ errorpage:

the error page attribute is used to define the error page if exception occurs in Current page, it will be redirected to error page.

Example: <%@ page errorpage="my errorpage.jsp" %>

⑨ is Error page:

the is Errorpage attribute is used to declare that the Current page is the error page.

Example: <% @ page is Error page = "true" %>

2) JSp include Directive:

the include directive is used to include the Contents of any resource it may be jsp, html, text file etc.

Advantage of include Directive is Code Reusability

Syntax: <% @ include file = "resource name" %>

Example: <% @ include file = "Sai.html" %>

3) JSp Taglib directive:

the jsp taglib directive is used to define a tag library that defines many tags. we use the tag Library Descriptor (tlD) file to define the tags.

Syntax: <% @ taglib uri = "uri of tag library" prefix = "prefix of tag lib" %>

Example:

<% @ taglib uri = "https://www.nslectures.com/tags" prefix = "mytag" %>

Implicit objects :

The implicit objects are predefined Variables used to access request and application data, these objects are used by Scripting elements.

Variable name	Class/Interface name	meaning	Sample methods
① application	javax.Servlet. ServletContext	this object provides resources Shared with in a web application	log( ) getServerinfo( )
② Config	javax.Servlet. Servlet Config	It helps in passing information to Servlet or JSP page during initialization	get init parameter() getServletName( )
③ request	javax.Servlet.http HttpServlet Request	It provides the method for Accessing information made by Current Request	get Contentlength() getlocal Address( ) get ServerName( )
④ out	javax..Servlet. jsp. JSP writer	It provides the method related to Info	clear() newline( )
⑤ response	javax.Servlet.http Http Servlet Response	It provides method related to adding Cookies, Sessions or Setting	addcookie( ) add header( ) flush buffer( ) get Content type( ) set Content type( )
⑥ page	java.lang. Object	this variable is assigned to instance of JSP implementation class. It's rarely used	
⑦ pageContext	javax.Servlet.jsp. page Context	It provides access to Serveral JSP page attributes	getpage(), get Request( ) get Response( ) get Session( )
8 Session	javax. Servlet.http Http Session	This Variable is used to access the Current client's session	get Id( ) get Creationtime( )