In JSP page to add Java code we use following symbols

<% java code %>	Scriplet-→ code inside scriplet gets added in the service method of servlet
<%!---- declarative%>	Jsp lifecycle methods are jspInit(), jspDestroy(), _jspService(), out of these we may override jspInit() and jspDestroy(), and then these methods has to be written inside <%!---declarative %>
<%@ %>	This symbol is used for directives, in jsp there are 3 directives, page, include, taglib
<%= %>	Expression, it is used to display value of a variable, or the value returned by any function

In JSP there are 3 directives

1. <%@page%>
   It gives extra information about page to compiler
   It has following attributes

buffer	Default value of buffer is 8kb, but you may change the buffer size incase needed <%@page buffer=20kb"%>
isErrorPage	In jsp page where we need access to exception object, then we use isErrorPage attribute to true myerrorpage.jsp <%@page isErrorPage="true" %>
errorPage	Assign the value of the jsp page, to which we want to pass the control if any error occurs in the current jsp page Myjsppage.jsp <%@page errorPage="myerrorpage.jsp"%>
language	Its value is "java"
session	By default its value is true, hence jsp allows us to use session object in page. If you don't want to allow using session object, then assign value of session attribute to false.
isELIgnored	If on the jsp page we are using EL language, and if its o/p is not appearing on the page, then we need to add the attribute isELIgnored="false"
import	Assign all import packages to import attribute <%@page import="java.util.Date,com.demo.beans.Product" %>

2. <%@include %>
   It is helpful to add output of one file at compile time in jsp, page,
   Usually useful for display same header, footer, logos on every page

3. <%@taglib%>
   It is used for JSTL(jsp standard template library)

   JSP actions

<jsp:useBean> </jsp:useBean>	It is used to create a object	<%Product p=new Product(12,"nachos");%> <jsp:useBean id="p" class="com.demo.beans.Product">
<jsp.setProperty></jsp:setProperty>	It calls setter method of the class	<jsp:setproperty name="p" property="pid" value="12"></jsp:setProperty> <jsp:setproperty name="p" property="*"></jsp:setProperty>
<jsp:getProperty></jsp:getProperty>	It calls appropariate getter method of the class	<jsp:getProperty name="p" property="pid"></jsp:getProperty>
<jsp:include></jsp:include>	It is similar to Requestdispatcher.include(request,response)	<jsp:include page="validatedata"></jsp:include>
<jsp:forward></jsp:forward>	It is similar to Requestdispatcher.forward(request,response)	<jsp:forward page="validatedata"></jsp:forward>
<jsp:param></jsp:param>	It is used to add extra parameters in request object	<jsp:param name="msg" value="Hello message"></jsp:param>

JSTL (jsp statndarad Template library)

1. Core <c:set> <c:out>
2. XML
3. Sql
4. Formatting

In core tag library tags are :

<c:set var="i" val="12"></c:set>	Assign value to variable i
<c:out val="${i}"></c:out>	To display value
<c:forEach var="I" begin="1" end="10" step="1"> <c:out val=${i}></c:out> </c:forEach>	To display values from 1 to 10
<c:forEach var="p" items="${plist}"> ${p.pid} //to display all product properties </c:forEach>	To display all product object from plist ${plist} → will automatically do (List<Product>)request.getAttribute("plist")
<u><c:choose></u>	Equivalent to switch case

<c:when test="${i==1}">Monday</c:when> <c:when test="${i==12}">Tuesday</c:when> <c:when test="${i==3}">Wednesday</c:when> <c:when test="${i==4}">Thursday</c:when> <c:otherwise>Friday</c:otherwise> </c:choose>	<c:otherwise> → this is equivalent to default case

Note : to use JSTL add standard.jar and jstl.jar file into lib folder

In jsp file add

<%@taglib prefix="c" uri="http://java.sun.com/jstl/core_rt" %>

Custom tags

1. Add custom tags in customtags.jsp page

```
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<%

  String fnm="Kishori";
        String lnm="Khadilkar";
%>
<h1>Customtag demo</h1>
<hello:myhello fname="<%=fnm%>" lname="<%=lnm %>"></hello:myhello>
<hello:calculate num1='<%=Integer.parseInt(request.getParameter("num1"))
%>' num2='<%=Integer.parseInt(request.getParameter("num2"))%>'>
</hello:calculate>
</body>
</html>
```

2. Add taglib entry in custometags.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
  pageEncoding="ISO-8859-1"%>
<%@taglib prefix="hello"  uri="/mytaglib"%>
<!DOCTYPE html>
<html>
```

3. Add following entry in web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" version="4.0">
 <display-name>CustomTagDemo</display-name>
 <welcome-file-list>
  <welcome-file>index.html</welcome-file>

 </welcome-file-list>
 <jsp-config>
 <taglib>
    <taglib-uri>/mytaglib</taglib-uri>
    <taglib-location>/WEB-INF/mytaglib.tld</taglib-location>
 </taglib>
 </jsp-config>
</web-app>
```

4. In WEB-INF add file mytaglib.tld (tld stands for (tag library definition)

```
<taglib>
 <tlib-version>1.0</tlib-version>
 <jsp-version>1.1</jsp-version>
 <tag>
  <name>myhello</name>
  <tag-class>com.demo.tags.HelloTagHandler</tag-class>
  <attribute>
   <name>fname</name>
   <required>true</required>
   <rtexprvalue>true</rtexprvalue>
  </attribute>
  <attribute>
   <name>lname</name>
   <required>true</required>
   <rtexprvalue>true</rtexprvalue>
  </attribute>

 </tag>
 <tag>
  <name>calculate</name>
  <tag-class>com.demo.tags.CalculateTagHandler</tag-class>
  <attribute>
   <name>num1</name>
   <required>true</required>
   <rtexprvalue>true</rtexprvalue>
  </attribute>
  <attribute>
   <name>num2</name>
   <required>true</required>
   <rtexprvalue>true</rtexprvalue>
  </attribute>

 </tag>
```

```
</taglib>
```

5. Add HelloTagHandler class in com.demo.tags package

```
package com.demo.tags;

import java.io.IOException;

import javax.servlet.jsp.JspWriter;
import javax.servlet.jsp.tagext.TagSupport;

public class HelloTagHandler extends TagSupport{
	private String fname,lname;

	public String getFname() {
		return fname;
	}

	public void setFname(String fname) {
		this.fname = fname;
	}

	public String getLname() {
		return lname;
	}

	public void setLname(String lname) {
		this.lname = lname;
	}

	public int doStartTag() {
		return EVAL_BODY_INCLUDE;

	}

	public int doEndTag() {
		JspWriter out=pageContext.getOut();
		try {
		out.println("Welcome to hello tag library");
		out.println("we are displaying fname and lname");
		out.println("Hello : "+fname+"  "+lname);
		}catch(IOException e) {
			e.printStackTrace();
		}
		return EVAL_PAGE;
	}


}
```

6. Add CalculateTagHandler class

```
	package com.demo.tags;

	import java.io.IOException;

	import javax.servlet.jsp.JspWriter;
	import javax.servlet.jsp.tagext.TagSupport;

	public class CalculateTagHandler extends TagSupport {
		private int num1,num2;

		public int getNum1() {
			return num1;
		}

		public void setNum1(int num1) {
			this.num1 = num1;
		}

		public int getNum2() {
			return num2;
		}

		public void setNum2(int num2) {
			this.num2 = num2;
		}
		public int doStartTag() {
			return EVAL_BODY_INCLUDE;
		}

		public int doEndTag() {
			JspWriter out=pageContext.getOut();
			try {
				out.println("this is calculate tag");
				out.println("attributes are: num1 -->"+num1+"  Num2 :
	"+num2);
				out.println("Addition : "+(num1+num2));
			} catch (IOException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			return EVAL_PAGE;
		}

}
```