JSP

(Java Server Pages)

**JSP** technology is used to create web application. It can be thought of as an extension to servlet because it provides more functionality than servlet such as **expression** **language**, **jstl** etc.

A JSP page consists of HTML tags and JSP tags

**Advantage of JSP**:

#### Extension to Servlet

#### Easy to maintain

#### Fast Development: No need to recompile and redeploy

#### Less code than Servlet

#### **Life cycle of a JSP Page:**

* Translation of JSP Page
* Compilation of JSP Page
* Classloading (class file is loaded by the classloader)
* Instantiation (Object of the Generated Servlet is created).
* Initialization ( jspInit() method is invoked by the container).
* Reqeust processing ( \_jspService() method is invoked by the container).
* Destroy ( jspDestroy() method is invoked by the container).

#### **Note:** jspInit(), \_jspService() and jspDestroy() are the life cycle methods of JSP.

JSP page is translated into servlet by the help of JSP translator. The JSP

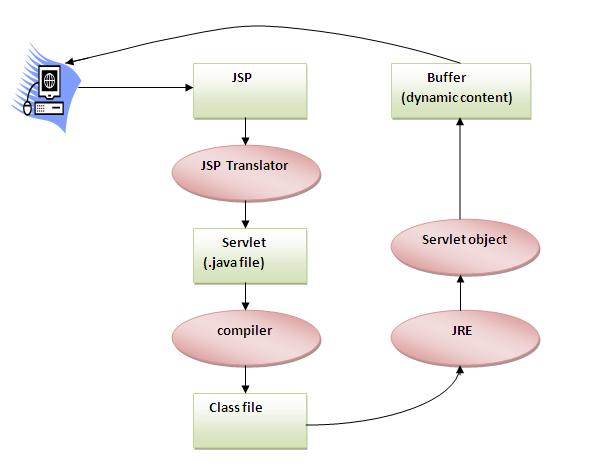
translator is a part of webserver that is responsible to translate the JSP

page into servlet. Afterthat Servlet page is compiled by the compiler and

gets converted into the class file. Moreover, all the processes that

happens in servlet is performed on JSP later like initialization, committing

response to the browser and destroy.



**Sample Code:**

<html>

<body>

<% out.print(10\*20); %>

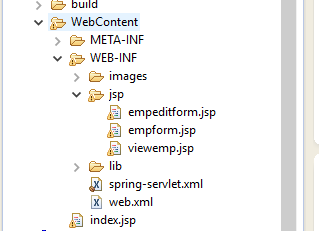
</body>

</html>

It will print **200** on the browser.

.jsp Pages in Eclipse or STS:

Directory Structure:



.jsp pages should be in WebContent-🡪 WEB\_INF -🡪 jsp -🡪 pages.jsp

# **The JSP API**

The JSP API consists of two packages:

1. javax.servlet.jsp
2. javax.servlet.jsp.tagext

# **JSP Scriptlet tag:**

The scripting elements provides the ability to insert java code inside the jsp.

By using

* scriptlet tag: <%  java source code %>
* expression tag: **<**%=  statement %**>**
* declaration tag: **<**%!  field or method declaration %**>**

# **JSP Implicit Objects:**

There are **9 jsp implicit objects**. These objects are *created by the web container*that are available to all the jsp pages.

Object Type Usage

1. out JspWriter PrintWriter out=response.getWriter();

<% out.print("Hello”); %>

1. request HttpServletRequest

String name=request.getParameter("uname");

1. response HttpServletResponse

response.sendRedirect("http://www.google.com");

1. config ServletConfig String driver=config.getInitParameter("dname");
2. Application ServletContext

String driver=application.getInitParameter("dname");

1. Session HttpSession session.setAttribute(“user”,name);
2. pageContext PageContext

pageContext.setAttribute("user",name,PageContext.SESSION\_SCOPE);

1. Page Object Object page=this;
2. Exception Throwable <%@ page isErrorPage="true" %>

# **JSP directives:**

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

* page directive: <%@ page attribute="value" %>  (We have attributes also)
* include directive: <%@ include file="resourceName" %>
* taglib:<%@ taglib uri="uriofthetaglibrary" prefix="prefixoftaglibrary" %>

# **JSP Action Tags:**

There are many JSP action tags or elements. Each JSP action tag is used to perform some specific tasks.

The action tags are used to control the flow between pages and to use Java Bean.

**jsp:forward :** forwards the request and response to another resource.

<jsp:forward page="relativeURL | <%= expression %>">

<jsp:param name="parametername" value="parametervalue | <%=expression%>" />

</jsp:forward>

**Jsp:include:** include another resource

<jsp:include page="relativeURL | <%= expression %>" />

**jsp:useBean:** creates or locates bean object.

<jsp:useBean id= "instanceName" scope= "page | request | session | application"

**class**= "packageName.className" type= "packageName.className"

beanName="packageName.className | <%= expression >" >

</jsp:useBean>

**jsp:setProperty:** Set the value of the property in the Bean object.

<jsp:setProperty name="instanceOfBean" property= "\*"   |

property="propertyName" param="parameterName"  |

property="propertyName" value="{ string | <%= expression %>}"

/>

**jsp:getProperty:** prints the value of property of the bean.

<jsp:getProperty name="instanceOfBean" property="propertyName" />

**jsp:plugin:** embeds another components such as applets.

**<jsp:plugin** type= "applet | bean" code= "nameOfClassFile"

codebase= "directoryNameOfClassFile"

**</jsp:plugin>**

**jsp:param:** sets the parameter value. It is used in forward and include mostly.

# **Expression Language (EL) in JSP:**

The **Expression Language** (EL) simplifies the accessibility of data stored in the Java Bean component, and other objects like request, session, application etc.

There are many implicit objects, operators and reserve words in EL.

${ expression }

**Reserve words in EL**

There are many reserve words in the Expression Language. They are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| lt | le | gt | ge |
| eq | ne | true | false |
| and | or | not | instanceof |
| div | mod | empty | null |

# **MVC in JSP**

**MVC** stands for Model View and Controller. It is a **design pattern** that separates the business logic, presentation logic and data.

**Controller** acts as an interface between View and Model. Controller intercepts all the incoming requests.

**Model** represents the state of the application i.e. data. It can also have business logic.

**View** represents the presentaion i.e. UI(User Interface).



# **JSTL (JSP Standard Tag Library):**

The JSP Standard Tag Library (JSTL) represents a set of tags to simplify the JSP development.

## **Advantage of JSTL:**

1. **Fast Developement** JSTL provides many tags that simplifies the JSP.
2. **Code Reusability** We can use the JSTL tags in various pages.
3. **No need to use scriptlet tag** It avoids the use of scriptlet tag.

**Note**: For creation of jstl application , you need to download **jstl.jar**