# JSP

**JSP** technology is used to create web application just like Servlet technology. 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. 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 Tag etc.

## Advantage of JSP over Servlet

There are many advantages of JSP over servlet. They are as follows:

### 1) Extension to Servlet

JSP technology is the extension to servlet technology. We can use all the features of servlet in JSP. In addition to, we can use implicit objects, predefined tags, expression language and Custom tags in JSP, that makes JSP development easy.

### 2) Easy to maintain

JSP can be easily managed because we can easily separate our business logic with presentation logic. In servlet technology, we mix our business logic with the presentation logic.

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

If JSP page is modified, we don't 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 the application.

### 4) Less code than Servlet

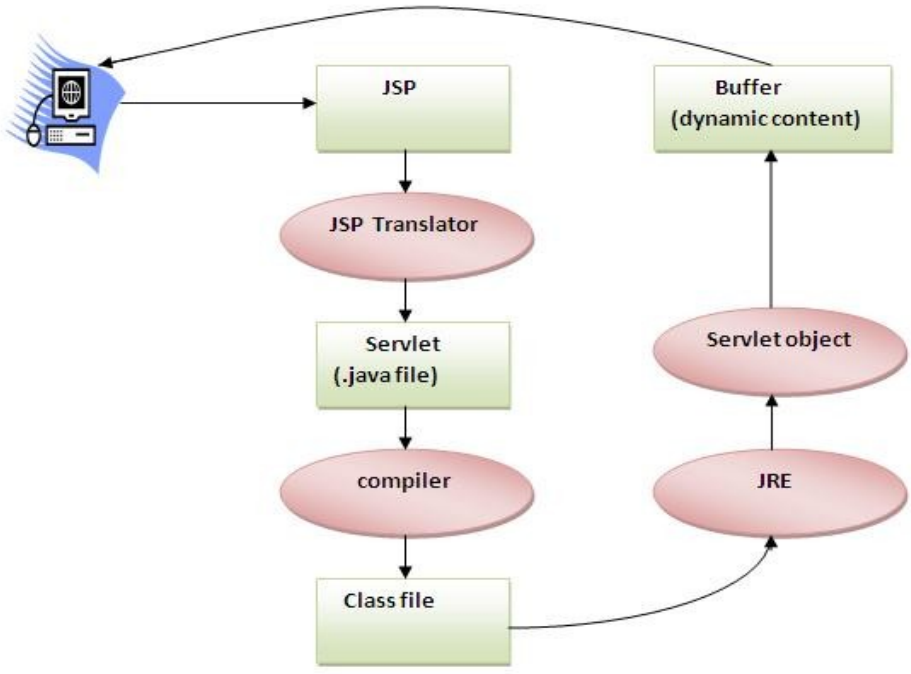
In JSP, we can use a lot of tags such as action tags, jstl, custom tags etc. that reduces the code. Moreover, we can use EL, implicit objects etc.

## Life cycle of a JSP Page

The JSP pages follows these phases:

- 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.***




As depicted in the above diagram, 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.

## What is JavaServer Pages?

JavaServer Pages (JSP) could be a technology for developing web content that support dynamic content that helps developers insert java code in HTML pages by creating use of special JSP tags, most of that begin with <% and finish with %>. A JavaServer Pages element could be a form of Java servlet that's designed to satisfy the role of a program for a Java internet application. internet developers write JSPs as text files that mix HTML or XHTML code, XML parts, and embedded JSP actions and commands. Using JSP, you'll be able to collect input from users through online page forms, gift records from a information or another supply, and build web content dynamically. JSP tags are often used for a range of functions, like retrieving data from a information or registering user preferences, accessing JavaBeans parts, passing management between pages and sharing data between requests, pages etc.

## Why Use JSP?

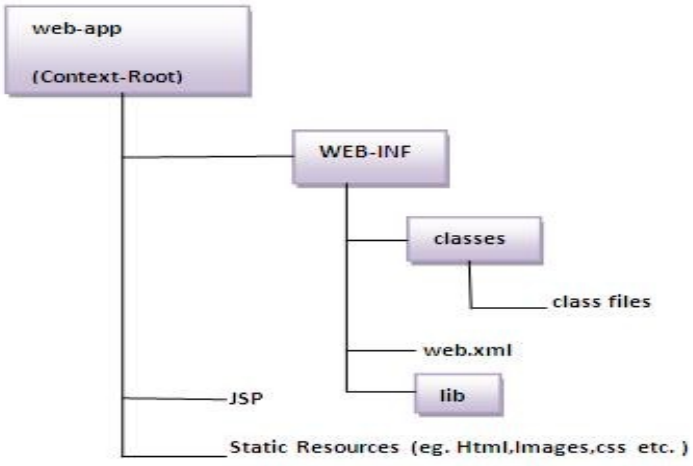
JavaServer Pages usually serve a similar purpose as programs enforced victimisation the Common entree Interface (CGI). however JSP provide many blessings compared with the CGI. Performance is considerably higher as a result of JSP permits embedding Dynamic parts in HTML Pages itself rather than having a separate CGI files. JSP area unit forever compiled before it's processed by the server not like CGI/Perl which needs the server to load AN interpreter and therefore the target script on every occasion the page is requested. JavaServer Pages area unit designed on high of the Java Servlets API, thus like Servlets, JSP conjointly has access to all or any the powerful Enterprise Java Apis, as well as JDBC, JNDI, EJB, JAXP etc. JSP pages are often employed in combination with servlets that handle the business logic, the model supported by Java servlet guide engines. Finally, JSP is AN integral a part of Java electrical engineering, a whole platform for enterprise category applications. this suggests that JSP will play an area within the simplest applications to the foremost advanced and tightened.

## Creating a simple JSP Page

To create the first jsp page, write some html code as given below, and save it by .jsp extension. We have save this file as test1.jsp. Put it in a folder and paste the folder in the web-apps directory in apache tomcat to run the jsp page.

## Directory structure of JSP

The directory structure of JSP page is same as servlet. We contains the jsp page outside the WEB-INF folder or in any directory.



## How to run a simple JSP Page ?

Create new project inside webapps

webapps --> basic-sciptlets (New Project).

### Program 1:

Develop test1.jsp in

webapps ---> basic-sciptlets ---> test1.jsp

**test.jsp**

```
Welcome to JSP
```

Start the Server,

Open the Web browser, and type,

http://localhost:7070/basic-sciptlets/test.jsp

**Output:**

Welcome to JSP

### Program 2:

Develop test2.jsp in

webapps ---> basic-sciptlets ---> test2.jsp

**test2.jsp**

```
<b>Welcome to JSP</b> <br>
<i>Welcome to JSP</i> <br>
<u>Welcome to JSP</u> <br>
<u><b>Welcome to JSP </b></u>
```

Start the Server,

Open the Web browser, and type,

http://localhost:7070/basic-sciptlets/tes2jsp

**Output:**

**Welcome to JSP**
*Welcome to JSP*
Welcome to JSP
**Welcome to JSP**

## JSP Tags:

We cannot keep java code directly in JSP.
In JSP, java code can be written inside the jsp page using the scriptlet tag.
Let's see what are the scripting elements first.

## There are 3 types of scripting elements:

The scripting elements provides the ability to insert java code inside the jsp. There are three types of scripting elements:

1. Scriptlet tag
2. Expression tag
3. Declaration tag

### 1. Scriptlet:

- Syntax : <% java source code %>
- Inside Scriptlet, we're keeping the java statements.
- Scriptlets are almost look like a statements which are inside a method. Whatever statements we're keeping inside a method, same statements, we can keep inside a Scriptlet.
  Whatever statements we're keeping inside a Scriptlet, those should end with a semicolon (;)
- Any no. of statements we can keep inside a Scriptlet.
- Any no. of Scriptlets we can keep in a JSP File.

### 2. Expression:

- Syntax : <%= statement %>
- Expression is used to print all Literals, Method return values, & variable values.
- Whatever u like to print, same thing we can keep inside a Expression
- The code placed within expression tag is written to the output stream of the response. So you need not write out.print() to write data. It is mainly used to print the values of variable or method.
- we cannot keep plain java code.
- Whatever statements we're keeping inside a Scriptlet, those should not end with a semicolon (;)

### 3. Declaration:

- Syntax : <%! field or method declaration %>
- 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 auto generated servlet.

  So it doesn't get memory at each request.
- The JSP declaration tag is used to declare global members (global variables & definine methods)
- In 'Declaration tag', we're declaring global variables & we're defining entire methods. But we're not defining part of a method, (i.e) we're not keeping only method statements. We're developing entire method itself. Where as in "Scriptlet tag", we're defining part of method (i.e.) only statements which are defined like a inside method.
- In 'Declaration tag', we can define entire method.

**Difference between the jsp scriptlet tag and jsp declaration tag?**

| Jsp Scriptlet Tag | Jsp Declaration Tag |
|---|---|
| The jsp scriptlet tag can only declare variables not methods. | The jsp declaration tag can declare variables as well as methods. |
| The declaration of scriptlet tag is placed inside the _jspService() method. | The declaration of jsp declaration tag is placed outside the _jspService() method. |

## 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:

- page directive
- include directive
- taglib directive
- import
- contentType
- extends
- info
- buffer
- language
- isELIgnored
- isThreadSafe
- autoFlush
- session
- pageEncoding
- errorPage
- isErrorPage

## 1. Scriptlet tag:

**Program 3:**

**test3.jsp**

```
<%
	int i = 100;
	i++;
%>

i value is : <%= i  %>
```

**Output:**

```
i value is : 101
```

## 2. Expression tag:

**Program 4:**

**test4.jsp**

```
<%
	int i = 100;
	i++;
%>

<%
	int j = i;
	j += 200;
%>

i value is : <%= i  %> <br>
j value is : <%= j %>
```

Output:

```
i value is : 101
j value is : 301
```

**Explanation:**

Here, we developed 2 Scriptlets.

**Program 5:**

**test5.jsp**

```
<%=   1000        %> <br>
<%=   10.105f     %> <br>
<%=   1000.1005   %> <br>
<%=   true        %> <br>
<%=   false       %> <br>
<%=   'c'         %> <br>
<%=   "HELLO"     %> <br>
<%=   "Hello " + " Buddy" %> <br>
<%=   100+200     %> <br>
<%=   100*200     %>
```

**Output:**

```
1000
10.105
1000.1005
true
false
c
HELLO
Hello Buddy
300
20000
```

**Explanation:**

We're Printing various types of literals %& Expressions.

**Program 6:(Multiple Scriptlets with Multiple Expressions)**

**test6.jsp**

```
<%
      int i =100;
      i+= 10;
%>

<%=   "i value is  :   "  +i   %>  <br>

<%
      i+= 20;
      i++;
%>

<%=   "i value is  :  " +i  %>      <br>

<%
      i+=   30;
      i++;
%>

i value is  :  <%=   i  %>
```

**Output:**

```
i value is : 110
i value is : 131
i value is : 162
```

**Program 7:**

**test7.jsp**

```
<%
	java.util.Date date = new  java.util.Date();
	String s = "Sashi";
%>

Date is : <%= date %> <br>
s value : <%= s  %>
```

**Output:**

```
Date is : Wed Apr 16 09:00:37 IST 2014
s value : Sashi
```

## 3. Declaration tag:

**Program 8:**

**test8.jsp**

```
<%!
	int i = 100;
	void test()
	{
	}

%>
i value is  :  <%=  i  %>
```

**Output:**

```
i value is : 100
```

**Explanation:**

Here, in 'Declaration tag', we're declaring i variable. 'i' is a global variable. And we're defining a test() method. test() method also global method

**Program 9:**

**test9.jsp**

```
<%!
	float f = 200.2f;
	float test()
	{
		return (100+f);
	}
%>

f value is : <%=  f  %>  <br>
test() is :  <%=  test()  %>
```

**Output:**

```
f value is  : 200.2
test() is    : 300.2
```

**Program 10:**

**test10.jsp**

```
<%!
      String s ="Global variable";
%>

S value is : <%=
      s
%>
```

**Output:**

```
S value is : Global variable
```

**Program 11:**

**test11.jsp**

```
<%!
      String s ="Global variable";
%>

<%
      String s = "Local variable";
%>
S value is :      <%=
                        s
                  %>
```

**Output:**

```
S value is : Local variable
```

**Explanation:**

Here, in 'Declaration tag', we're declared global variable 's'.
same name 's' is declared in "Scriptlet" also.

Then which one will get more preference??
Question : Global variable or Local variable?
Answer   : Local variable only will get more preference.

**Program 12:**

Question : How to access global variable?
Answer : Use 'this' keyword to access global variables.

**test12.jsp**

```
<%!
      String s ="Global variable";
%>

<%
      String s = "Local variable";
%>

Global S value is : <%= this.s  %> <br>
Local  S value is : <%= s  %>
```

**Output:**

```
Global S value is : Global variable
Local S value is : Local variable
```

**Explanation:**

Here, in 'Declaration tag', we're declared global variable 's'.
same name 's' is declared in "Scriptlet" also.

Local variable is getting more preference if u access directly by using variable name (i.e.) s.

If u like to access global variable 's', then use 'this' keyword.
(i.e.)this.s

**Program 13:**

**test13.jsp**

```
<%!
      int day = 2;
%>

<%
      if(day == 1)  { %>
            1st Day
      <% } else if (day ==2) { %>
            2nd Day
      <% } else  { %>
            Some other Day
<% } %>
```

**Output:**

```
2nd Day
```

**Program 14:**

**test14.jsp**

```
<%    int day =3; %>

Switch Statement is used. <br>

<%
      switch(day) {
            case  1:
                  out.println("Day 1");
                  break;
            case  2:
                  out.println("Day 2");
            break;
            case  3:
                  out.println("Day 3");
            break;
            default:
                  out.println("Some other day");
      }
%>
```

**Output:**

Switch Statement is used.
Day 3

**Program 15:**

**test15.jsp**

```
For Loop Example <br> <br>

<%!
      int fontSize;
%>

<%    for (fontSize = 1; fontSize<3; fontSize++) { %>
            <font color = "Red"> <%=  fontSize  %>
            JSP Tutorial <br>
<% }  %>
```

**Output:**

For Loop Example

1 JSP Tutorial
2 JSP Tutorial

## Directive tag:

**Program 16:**

**test16.jsp**

```
<%@ page import = "java.util.Date"  %>

<%
      Date d = new Date();
%>

<%= "Today Date is : " %>
<%= d %>
```

**Output:**

2nd Day