JSP (Java Server pages)

Drawbacks of servlet technologies:-

① whenever we are developing the servlet must and should we have to configure inside the web.xml file.

② Whenever we modifying the servlet must and should we need to stop the server and we need to compile the servlet and once again redeploye the application inside the server. and restart the server.

③ Servlets are allowed by the only Java code. but not text and HTML code.

④ ~~presenting~~
whenever we are using the servlets presenting the data is very slow.

$\boxed{\text{JSP techonolgy is given by the sun microsystem}}$

⇒ by using servlet and JSP we can developing dynamic web pages.

advantage of jsp technologies:-

① Whenever we are developing the one JSP page we no need to configure inside the web.xml file.

② presenting the data is very fast compare to servlets.

③ Whenever we are modifying the jsp we no need to stop the server and restart the server.

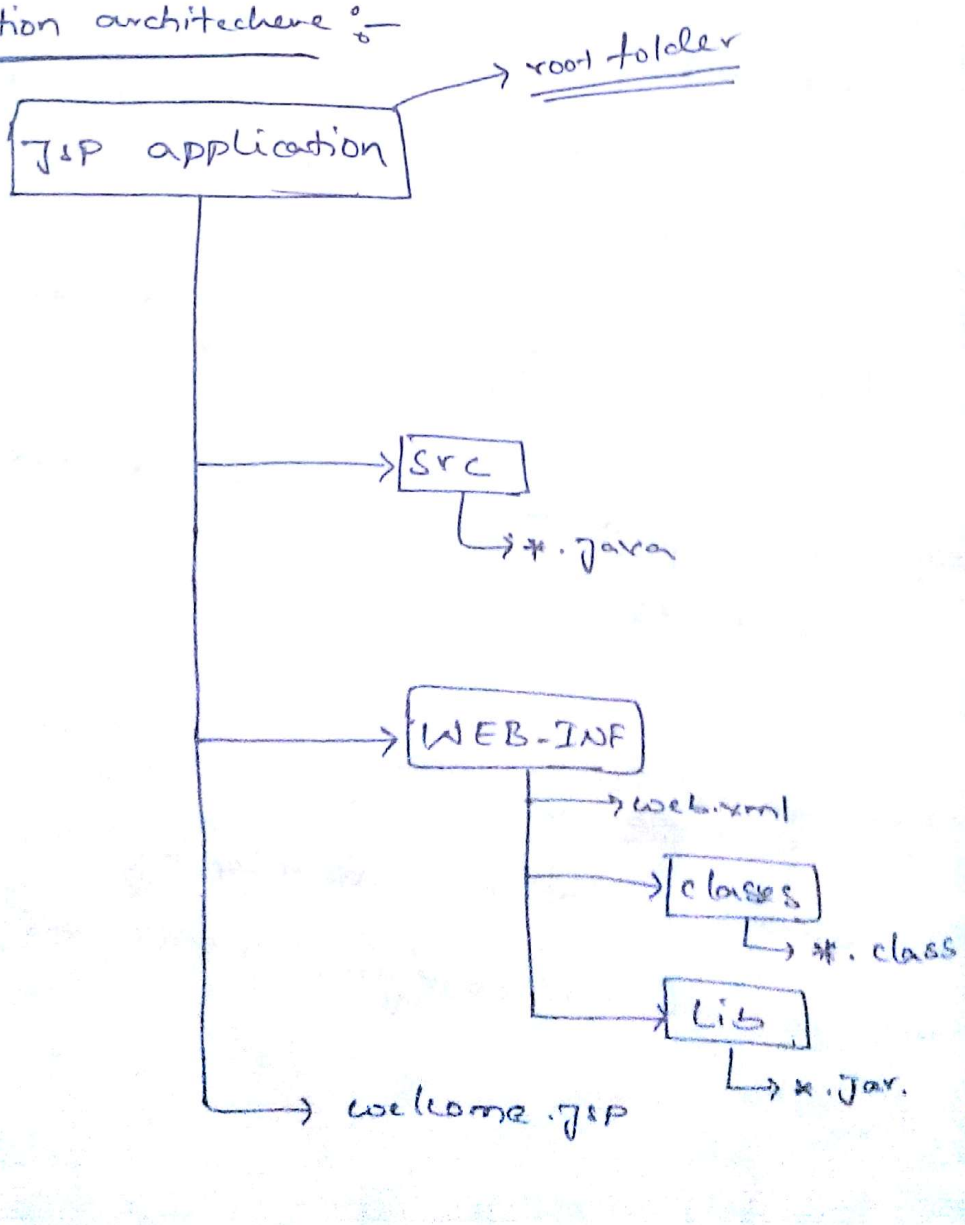
④ The JSP pages is allowed by the HTML code & Textual data & Java code also.

Rules of JSP page

① Whenever we are developing one JSP page, must and should we need to save extension is .JSP.

② after developing the jsp page must and should we need to place inside the application scope.

web application architechure :-



JSP elements

JSP elements →

① Scripting elements
- (i) declarative element
- ii) scriptlet element
- iii) expression element

② directives in JSP
- (i) page directive
- ii) include directive
- iii) taglib directive

② JSP action tags

① JSP predefined action tags
- i) forward action tag
- ii) include action tag
- iii) param action tag
- iv) use Bean action tag
- v) set property action tag
- vi) get property action tag

② customtag (we have to define custom tags)

with out having the JSP elements we can't write the Java code in JSP. file.

① declarative element

by using this element we can declare instance variable & methods

syntax

```
<%!
declare the instance
variable and method
        %>
```

Ex:-

```
<%!
 int count = 0;
void sum A()
   {
 - - - -
   }
     %>
```

② Scriptlet element :-

By using this element we can write some business logic.

syntax:-

```
<%
implementing the
business logic
   %>
```

Ex:-

implicit object ↑

```
<%
int a = integer.parseInt(request.
 getParameter("t1"));
int b = integer.parseInt(request.
 getParmeter("t2"));
int result = a+b;
 %>
```

③ Expression Element :-

By using Expression Element we can print the Result.

syntax :-

```
<%= expression %>
```

Ex :-

```
<% =a+b %>

     (or)

<%= result %>
```

now let me want to know how many times client is hitting the server side application. so here we have to create One application.

JSP hit application

step②

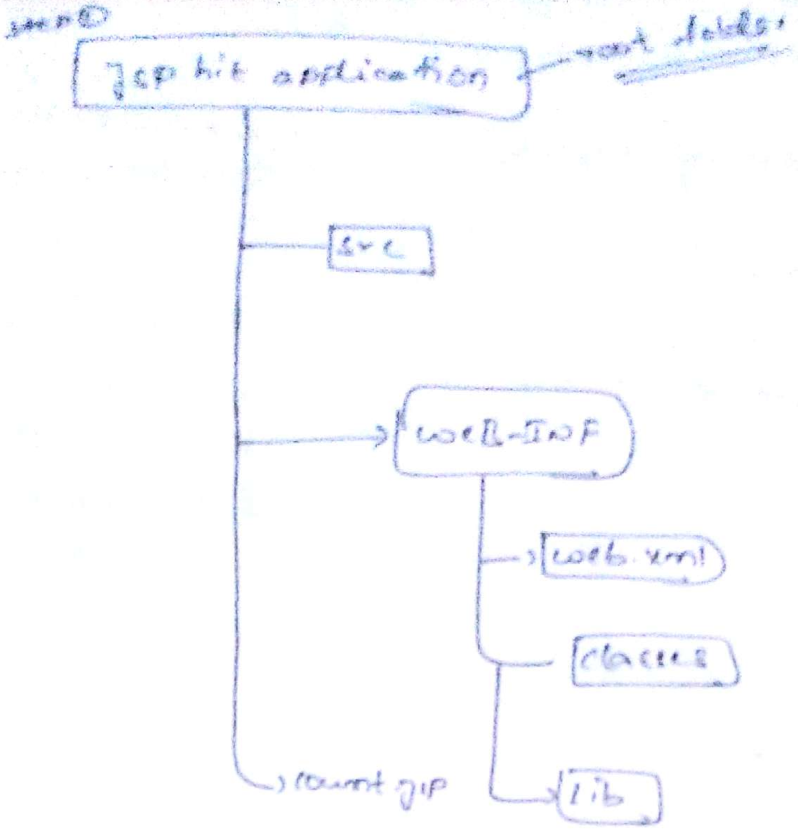
```
<!-- count.jsp --->
  <html>
  < body colour := "yellow" >
   < center >
   <%!
      Int count:0 %>          -> declarative element
   <br> <br>
   < % count++
           %>                 -> scriptlet element
     <br> <br>
< h1 The number of times client was hitting To server >
     <br> <br>
     <%= count %>             -> Expression element .
       <br><br >
       < /center >
       < /body>
       </html>
```

web.xml

```
<web.app>
</web app>
```

Scanned by CamScanner

step②



step③
Now we can deploy in

Tomcat
└→ webapp
  └→ jsp hit application

Step④: start the server

Step⑤: make a servlet request

http:// localhost: 2014/ jsphitapplication /count.jsp

Step ⑥ : 1 while refreshing count will increase.
2

Like this we can develop the applications. Compare to servlets jsp is very simple in developing dynamic web applications.

JSP

JSP	servlet
① request	javax.servlet.http. Servlet request.
② Responde.	javax. servlet. http. Servlet response.
③ Config	javax. servlet. servlet config
④ Exeption	java. lang. Exeption
⑤ page	java. lang. object.
⑥ pagecontext	java. servlet. jsp. Jspwriter
⑦ out	java. io. printwriter
⑧ session	javax. servlet. HTTP session
⑨ Application	javax. servlet. Servlet Context

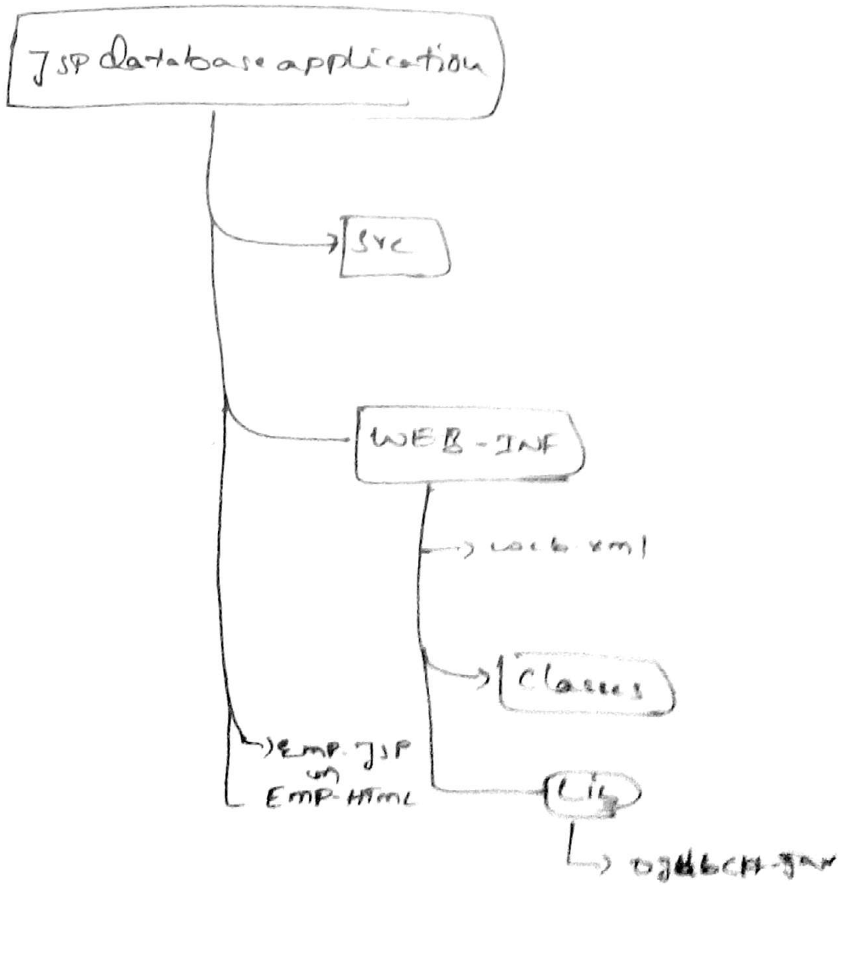
These are all 9 implicit objects

As part of JSP's there are 3 Directives are there.

As part of JSP cycle cycle there are 3 life cycles are available

JSP		Servlet
① jspinit() (one time executable code)	→	init()
② jspDestroy()	(destroy an object) →	destroy()
③ _jspService()	→	Service()

_jspService() ↓ business logic going to be executed

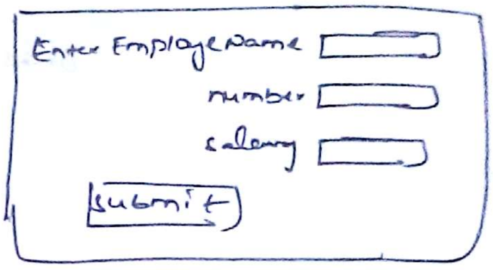


employee.html client side code

Browser

```
<html>
<body bgcolour = "yellow" >
< center >
< form action = "emp.jsp" method = "post" >
EnterEmployeenumber : <input type = "text" name = "eno> <br>
Enter Employee name = < input type = "text" name = "name" ><br>
Enter Employee salary = < input type = "text" name = "salary"><br>
<input type = "submit" value "send" >

   </form >
   </Center>
   </ body>
   </HTML>
```

Employee.html



server side code

Tomcat server

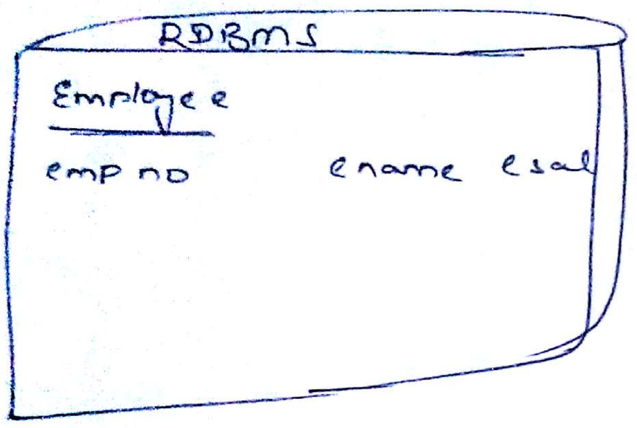
```
<%@ page import = "javax.sql.*"%>

<%!
      connection con=null;
      prepared statement pstmt = null;

   public void jspinit() throws JspException
      {     try{
   class.forName ("oracle.JDBC.driver.oracledriver");
   Con: DriverManager.get Connection ("JDBC:oracle:thin:
          @localhost:1521:XE" durgatech", "durgatech");

  pstmt= con.prepare statement("insert into Employe values
                                    (???)");
         } catch (Exeption e)
            {
            e. print stackTrace();
            }

         }
```

```
public void jspDestroy() Throws JSPException
    {
     try
       {
        if (pstmt! = null)
          pstmt. close();
        if (con ! = null)
             con. close();
           }
           Catch (Exeption (c)
               {
             c. print Stack Trace();
              }
             }


<%
 int  empno = integer .parse int (request. getparameter("eno"));

 String ename = request. getparameter ("ename");

   float  esal = float. parseFloat (request. getParameter("esal"))

     -  -  -  -  -  -  -  -  -  -  -

     pstmt.  setInt ('1, empno);

     pstmt . set string (2, ename);
      pstmt.  setFloat (3. esal);
       int k = pstmt. exicuteUpdate ();
       out.println ("Record is inserted successfully");

          %>

     <%@ include file= "emp.Html" %>
```

web.xml

```
<web-app>
 <welcome-file-list>
 <welcome-file>emploee.html</welcome-file>
    </welcome-file-list>
    </web-app>
```

⇓



database

out logic is going in 3 serveres base on that we can say it is 3-tier application.

JSP Actiontags

① include action tag
② forward action tag
③ param action tag
④ setproperty action tag
⑤ get property action tag
⑥ useBean action tag
⋮
etc.

<JSP: tagname/>