Dt : 15/12/2021(Day-25)

Assignment:(Solution)

login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<form action="login" method="post">
UserName:<input type="text" name="uname"><br>
PassWord:<input type="password" name="pword"><br>
<input type="submit" value="Login">
<a href="Register.html">NewUser?</a>
</form>
</body>
</html>
```

link.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<a href="view">ViewProfile</a>
<a href="logout">Logout</a>
</body>
</html>
```

UserBean.java

```
package test;
import java.io.*;
@SuppressWarnings("serial")
public class UserBean implements Serializable{
   private String uName,pWord,fName,lName,addr,mId;
   private long phNo;
   public UserBean(){}
public String getuName() {
```

```
	return uName;
}
public void setuName(String uName) {
	this.uName = uName;
}
public String getpWord() {
	return pWord;
}
public void setpWord(String pWord) {
	this.pWord = pWord;
}
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 String getAddr() {
	return addr;
}
public void setAddr(String addr) {
	this.addr = addr;
}
public String getmId() {
	return mId;
}
public void setmId(String mId) {
	this.mId = mId;
}
public long getPhNo() {
	return phNo;
}
public void setPhNo(long phNo) {
	this.phNo = phNo;
}

}
```

DBConnection.java

```
package test;
import java.sql.*;
public class DBConnection {
   private static Connection con=null;
   private DBConnection(){}
   static{
         try{
               Class.forName("oracle.jdbc.driver.OracleDriver");
               con = DriverManager.getConnection

("jdbc:oracle:thin:@localhost:1522:orcl","system","manager");
         }catch(Exception e){e.printStackTrace();}
   }//end of static block
   public static Connection getCon(){
         return con;
   }
}
```

LoginDAO.java

```
package test;

import java.sql.*;

import javax.servlet.http.*;

public class LoginDAO {

  public UserBean ub=null;

  public UserBean login(HttpServletRequest req){

        try{

              Connection con = DBConnection.getCon();

              PreparedStatement ps = con.prepareStatement

                          ("select * from UserReg37 where uname=? and pword=?");

              ps.setString(1,req.getParameter("uname"));

              ps.setString(2,req.getParameter("pword"));
```

```
                ResultSet rs = ps.executeQuery();
                if(rs.next()){
                        ub=new UserBean();
                        ub.setuName(rs.getString(1));
                        ub.setpWord(rs.getString(2));
                        ub.setfName(rs.getString(3));
                        ub.setlName(rs.getString(4));
                        ub.setAddr(rs.getString(5));
                        ub.setmId(rs.getString(6));
                        ub.setPhNo(rs.getLong(7));
                }
        }catch(Exception e){e.printStackTrace();}
        return ub;
  }
}
```

LoginServlet.java

```
package test;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;
@SuppressWarnings("serial")
@WebServlet("/login")
```

```
public class LoginServlet extends HttpServlet{

        @Override

   protected void doPost(HttpServletRequest req,HttpServletResponse res)throws

   ServletException,IOException{

        UserBean ub = new LoginDAO().login(req);

        if(ub==null){

                req.setAttribute("msg","Invalid Login process...");

                req.getRequestDispatcher("Fail.jsp").forward(req,res);

        }else{

                HttpSession hs = req.getSession();

                hs.setAttribute("ub",ub);

                req.getRequestDispatcher("Success.jsp").forward(req,res);

        }

  }

}
```

Fail.jsp

```
<%@ page language="java"
    contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"
    %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1">
<title>Insert title here</title>
</head>
<body>
<%
  String msg = (String)request.getAttribute("msg");
```

```
  out.println(msg+"<br>");
%>
<jsp:include page="login.html"/>
</body>
</html>
```

Success.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"
    import="test.UserBean"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1">
<title>Insert title here</title>
</head>
<body>
<%
  UserBean ub = (UserBean)session.getAttribute("ub");
  out.println("Welcome User "+ub.getfName()+"<br>");
%>
<jsp:include page="link.html"/>
</body>
</html>
```

ViewProfileServlet.java

```
package test;

import java.io.*;

import javax.servlet.*;

import javax.servlet.http.*;

import javax.servlet.annotation.*;

@SuppressWarnings("serial")

@WebServlet("/view")

public class ViewProfileServlet extends HttpServlet{

        @Override
```

```
 protected void doGet(HttpServletRequest req,HttpServletResponse res)throws
 ServletException,IOException{
        HttpSession hs = req.getSession(false);//Access the existing Session
        if(hs==null){
            req.setAttribute("msg","Session Expired...");
            req.getRequestDispatcher("Fail.jsp").forward(req,res);
        }else{
            req.getRequestDispatcher("ViewProfile.jsp").forward(req,res);
        }
 }
}
```

LogoutServlet.java

```
package test;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;
@SuppressWarnings("serial")
@WebServlet("/logout")
public class LogoutServlet extends HttpServlet{
    @Override
 protected void doGet(HttpServletRequest req,HttpServletResponse res)throws
 ServletException,IOException{
        HttpSession hs = req.getSession(false);
```

```
        if(hs==null){
                req.setAttribute("msg","Session Expired...");
        }else{
                req.setAttribute("msg","User LoggedOut Successfully...");
        }
        req.getRequestDispatcher("Fail.jsp").forward(req,res);
 }
}
```

Assignment:

Update above application wil New User registration process.

Dt : 16/12/2021(Day-26)

Expression Language(EL) in JSP:

=>This EL simplifies the accessibility of data stored in java Bean component and other objects like request,session,application,etc..

Note:

It is newly added feature in JSP technology.

syntax of EL:

$(expression)

The following are the implicit objects of EL:

pageScope : It maps the given attribute name with the value,set in the

page scope

requestScope : It maps the given attribute name with the value set

inthe request scope

sessionScope : It maps the given attribute name with the value set

inthe session scope

applicationScope : It maps the given attribute name with the value set

inthe application scope

param : It maps the request parameter to the single value

paramValues :

It maps the request parameters to the array of values

header : It maps the request header name to the single value

headerValues : It maps the request header names to the array of values

cookie :It maps the cookie name to the cookie value

initParam : It maps the Initialization parameter

pageContext : It provides access to many objects request,session,...

==========

Summery of Objects Generated:

CoreJava Objects:

1.UserDefined Class objects

2.WrapperClass objects

(i)Byte Object

(ii)Short Object

(iii)Integer object

(iv)Long Object

(v)Float Object

(vi)Double Object

(vii)Character object

(viii)Boolean Object

3.String Objects

(i)String Object

(ii)StringBuffer object

(iii)StringBuilder object

4.Array Objects

(i)User defined Class Array

(ii)String Array

(iii)WrapperClass Array

(iv)Object Array

5.Collection<E> Objects

(a)Set<E>

(i)HashSet<E> object

(ii)LinkedHashSet<E> Object

(iii)TreeSet<E> Object

(b)List<E>

(i)ArrayList<E> Object

(ii)Vector<E> Object

|->Stack<E> Object

(iii)LinkedList<E> Object

(c)Queue<E>

(i)PriorityQueue<E> Object

|->Deque<E>

(ii)ArrayDeque<E> Object

6.Map<K,V> Objects

(i)HashMap<K,V> Object

(ii)LinkedHashMap<K,V> Object

(iii)TreeMap<K,V> Object

(iv)Hashtable<K,V> Object

7.Enum<E> objects

JDBC Objects:

1.Connection Object

2.Statement Object

3.PreparedStatement Object

4.CallableStatement object

5.Scrollable ResultSet Object

6.Non-Scrollable ResultSet Object

7.DatabaseMetaData object

8.ParameterMetaData object

9.ResultSetMetaData object

10.RowSet Object

11.Connection pooling object(Vector<E> Object)

Servlet Objects:

1.ServletContext object

2.ServletConfig Object

3.ServletRequest object/HttpServletRequest object

4.ServletResponse object/HttpServletResponse object

5.PrintWriter object

6.HttpSession object

7.Cookie object

8.DAO Layer object

9.JavaBean object

10.JCF Object

JSP objects:

1.application

2.config

3.request

4.response

5.out

6.session

7.exception

8.page

9.pageContext

JSP EL objects:

1.pageScope

2.sessionScope

3.requestScope

4.applicationScope

5.param

6.paramValues

7.cookie

8.pageContext

9.header

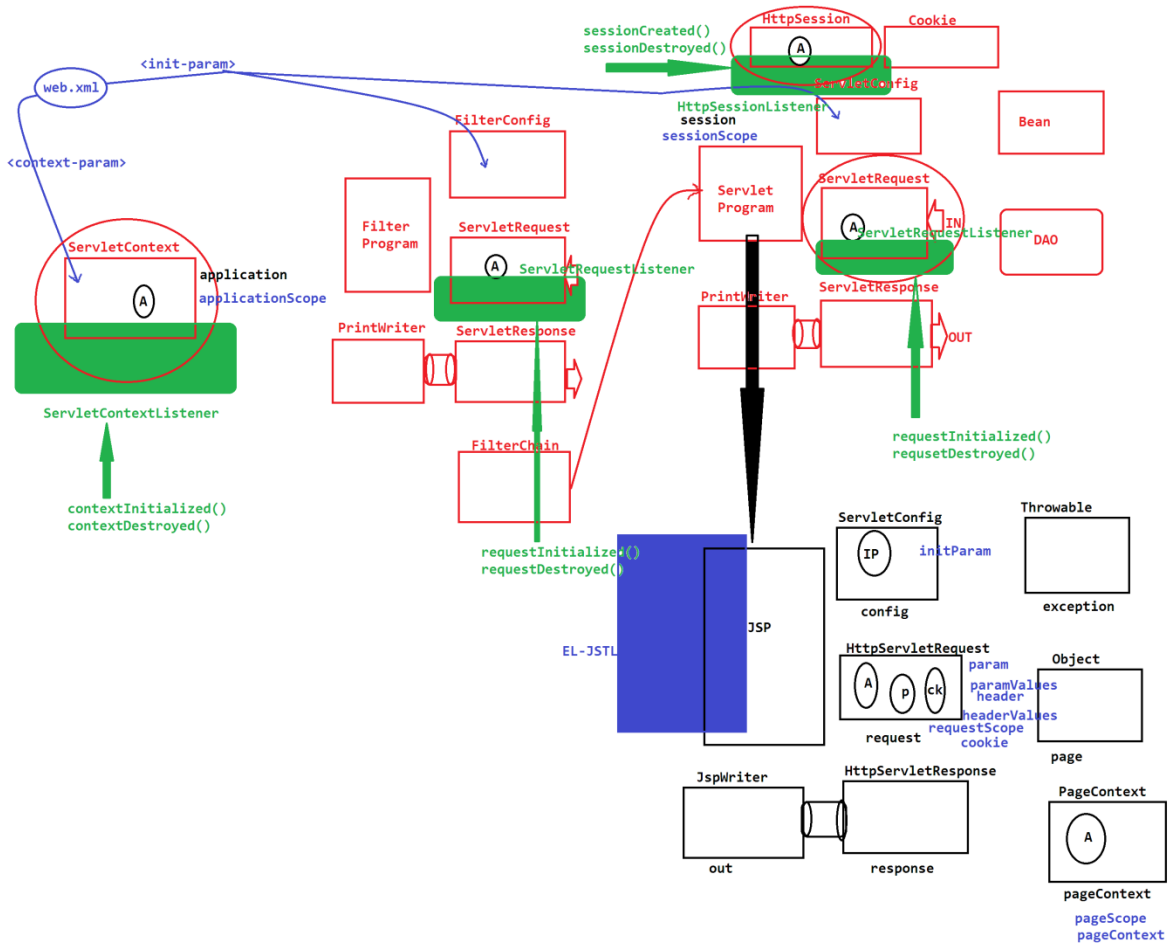
10.headerValues

11.initParam

Note:

=>In realtime JSPEL replaces with SpEL.

Objects Layout diagram:



==========

JSTL(JSP Standard Tag Lib):

=>This JSTL represents a set of tags to simplify the JSP development.

Advantages of JSTL:

(i)Fast Development

(ii)Code Reusability

(iii)No need to use scriptlet tag

The following are the JSTL tags:

(1)Core Tags

(2)Function Tags

(3)Formatting Tags

(4)XML Tags

(5)SQL Tags

Note:

=>we use "taglib" directive tag to declare JSTL Tags.

=>To Execute JSTL Tags we use the following steps:

(i)Download the following Jar files to execute JSTL tags

javax.servlet.jsp.jstl-1.2.1.jar

javax.servlet.jsp.jstl-api-1.2.1.jar

(ii)These jars must be available within 'lib' folder of

'WEB-INF' in deployment directory

---

(1)Core Tags:

=>These JSTL core tags provides variable support,URL management,

flow control etc. (Basic code writing)

syntax:

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

The following are the List of JSTL Core Tags:

c:out ->It displays the result of an expression, similar to

<%=...%> tag.

c:import->It Retrives relative or an absolute URL.

c:set-> It sets the value to variable.

c:remove->It is used for removing the variable .

c:catch-> It is used for Catching any Throwable exception that occurs

in the body.

c:if-> It is an conditional tag

c:choose, c:when, c:otherwise->

It is the simple conditional tag that includes its body content

if the evaluated condition is true.

**imp*

c:forEach-> It is the basic iteration tag.

c:forTokens-> It iterates over tokens which is separated by the

supplied delimeters.

c:param-> It adds a parameter in a containing 'import' tag's URL.

c:redirect-> It redirects the browser to a new URL and supports the context-relative URLs.

c:url-> It creates a URL with optional query parameters.

----

(2)Function Tags:

=>These JSTL function tags provides a number of standard functions, most of these functions are common string manipulation functions.

syntax:

<%@taglib uri= "http://java.sun.com/jsp/jstl/functions"

prefix="fn" %>

List of Some Function tags:

fn:contains() : It is used to test if an input string containing the specified substring or not.

fn:containsIgnoreCase(): It is used to test if an input string contains the specified substring as a case insensitive way.

fn:endsWith() : ***It is used to test if an input string ends with the specified suffix.***

fn:indexOf(): It returns an index within a string of first occurrence of a specified substring.

fn:trim(): It removes the blank spaces from both the ends of a string.

fn:startsWith(): It is used for checking whether the given string is started with a particular string value or not.

fn:split(): It splits the string into an array of substrings.

fn:toLowerCase(): It converts all the characters of a string to lower case.

fn:toUpperCase(): It converts all the characters of a string to uppercase.

fn:substring(): It returns the subset of a string according to the given start and end position.

fn:length(): It returns the number of characters inside a string, or the number of items in a collection.

fn:replace(): It replaces all the occurrence of a string with another string sequence.

---

(3)Formatting Tags:

=>The formatting tags provide support for message formatting, number formating and date formatting etc.

=>These formatting tags are also used for internationalized web sites to display and format text,time,date and numbers.

syntax:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt"
                        prefix="fmt" %>
```

List of some Formatting tags:

fmt:parseNumber : It is used to Parse the string representation of a currency, percentage or number.

fmt:formatNumber : It is used to format the numerical value with specific format or precision.

fmt:parseDate : It parses the string representation of a time and

date.

fmt:setTimeZone : It stores the time zone inside a time zone configuration variable.

**imp*

fmt:formatDate : It formats the time and date using the supplied pattern and styles.

----

(4)XML Tags:

=>The JSTL XML tags are used for providing a JSP-centric way of manipulating and creating XML documents.

syntax:

<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x" %>

(5)SQL Tags:

=>The SQL tag library allows the tags to Interact with RDBMS (Relational Databases) such as Microsoft SQL Server, mySQL, or Oracle.

syntax:

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

Note:

=>In realtime we must not have JSP Centric XML and DB Connections,

because of this reason XML tags and SQL Tags are less used when

compared to other tags.

==========

faq:

define Custom Tags?

=>The tags which are defined by the programmer are known as Custom Tags or

User defined Tags.

The following are twio ways to use custom tag:

way-1:

<prefix:tagname attr1=value1...attrn=valuen/>

way-2:

<prefix:tagname attr1=value1...attrn=valuen>

body Code

</prefix:tagname>

==========