Class 1

JSP

====

JSP stands for Java Server Pages.

JSP is a dynamic web resource program which is used to develop web applications.

Limitations with Servlets

==========================

> To work with servlet strong java knowledge is required.

> It is not suitable for non-java programmers.

> It does not give implicit objects.

> Handling exceptions are mandatory.

> Configuration of each servlet program in web.xml file is mandatory.

> We can't maintain HTML code and Java code seperately.

Advantages of JSP

==================

> To work with JSP strong java knowlege is not required.

> It is suitable for java and non-java programmers.

> It supports tag based language.

> It allows us to work with custom tags and third party supplied tags.

> It gives 9 implicit objects.

> Handling exception is optional.

> Configuration of each JSP program in web.xml file is optional.

> We can maintain HTML code and Java code seperately.

> It contains all the features of Servlets.

First web application development having JSP program as web resource program

=============================================================================

Deployment Directory Structure

------------------------------

JspApp1

|

|---Java Resources

|

|---WebContent

|

|---ABC.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

ABC.jsp

-------

<center>

<h1>

Current Date and Time <br>

<%

java.util.Date date=new java.util.Date();

out.println(date);

%>

</h1>

</center>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>ABC.jsp</welcome-file>

</welcome-file-list>

</web-app>

Request url

-----------

http://localhost:2525/JspApp1/

Configuration of JSP program in web.xml

=======================================

Deployment Directory Structure

------------------------------

JspApp1

|

|---Java Resources

|

|---WebContent

|

|---ABC.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

ABC.jsp

-------

<center>

<h1>

Current Date and Time <br>

<%

java.util.Date date=new java.util.Date();

out.println(date);

%>

</h1>

</center>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<servlet>

<servlet-name>ABC</servlet-name>

<jsp-file>/ABC.jsp</jsp-file>

</servlet>

<servlet-mapping>

<servlet-name>ABC</servlet-name>

<url-pattern>/test</url-pattern>

</servlet-mapping>

</web-app>

Request url

-----------

http://localhost:2525/JspApp1/test

http://localhost:2525/JspApp1/ABC.jsp

How can we access our application only by using url pattern

=============================================================

We can make our application accessible by using url pattern not by using file name. If we place ABC.jsp file inside "WEB-INF" folder.

Deployment Directory Structure

------------------------------

JspApp1

|

|---Java Resources

|

|---WebContent

|

|

|---WEB-INF

|

|---ABC.jsp

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

ABC.jsp

-------

<center>

<h1>

Current Date and Time <br>

<%

java.util.Date date=new java.util.Date();

out.println(date);

%>

</h1>

</center>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<servlet>

<servlet-name>ABC</servlet-name>

<jsp-file>/WEB-INF/ABC.jsp</jsp-file>

</servlet>

<servlet-mapping>

<servlet-name>ABC</servlet-name>

<url-pattern>/test</url-pattern>

</servlet-mapping>

</web-app>

Request url

-----------

http://localhost:2525/JspApp1/test (valid)

http://localhost:2525/JspApp1/ABC.jsp (Invalid)

Note:

-----

Servlet container is used to execute servlet program.

JSP container is used to execute JSP program.

But JSP container can't execute JSP program directly. It takes the support of servlet container to execute JSP program.

Internally, for every JSP program one JES class will be created.

JES stands for Java Equivalent Servlet class.

JSP life cycle methods

=======================

We have three life cycle methods in JSP.

1) \_jspInit()

-------------

It is used for instantiation event.

This method will execute just before JES class object creation.

2) \_jspService()

-------------

It is used for request arrival event.

This method will execute when request goes to JSP program.

3) \_jspDestroy()

-----------------

It is used for destruction event.

This method will execute just before JES class object destruction.

Q) What is the difference between Servlets and JSP ?

Servlets JSP

----------- -------------

To work with servlet strong java knowlege is To work with JSP strong java knowledge is not

required. required.

It is not suitable for non-java programmers. It is suitable for java and non-java programmers.

It does not support tags. It supports tags.

It is faster. It is slower.

It does not give any implicit object. It gives 9 implicit objects.

Handling exception is mandatory. Handling exception is optional.

Configuration of each servlet program Configuration of each jsp program in web.xml

in web.xml file is mandatory. file is optional.

We can't maintain HTML code and java code We can maintain HTML code and java code

seperately. seperately.

Phases in JSP

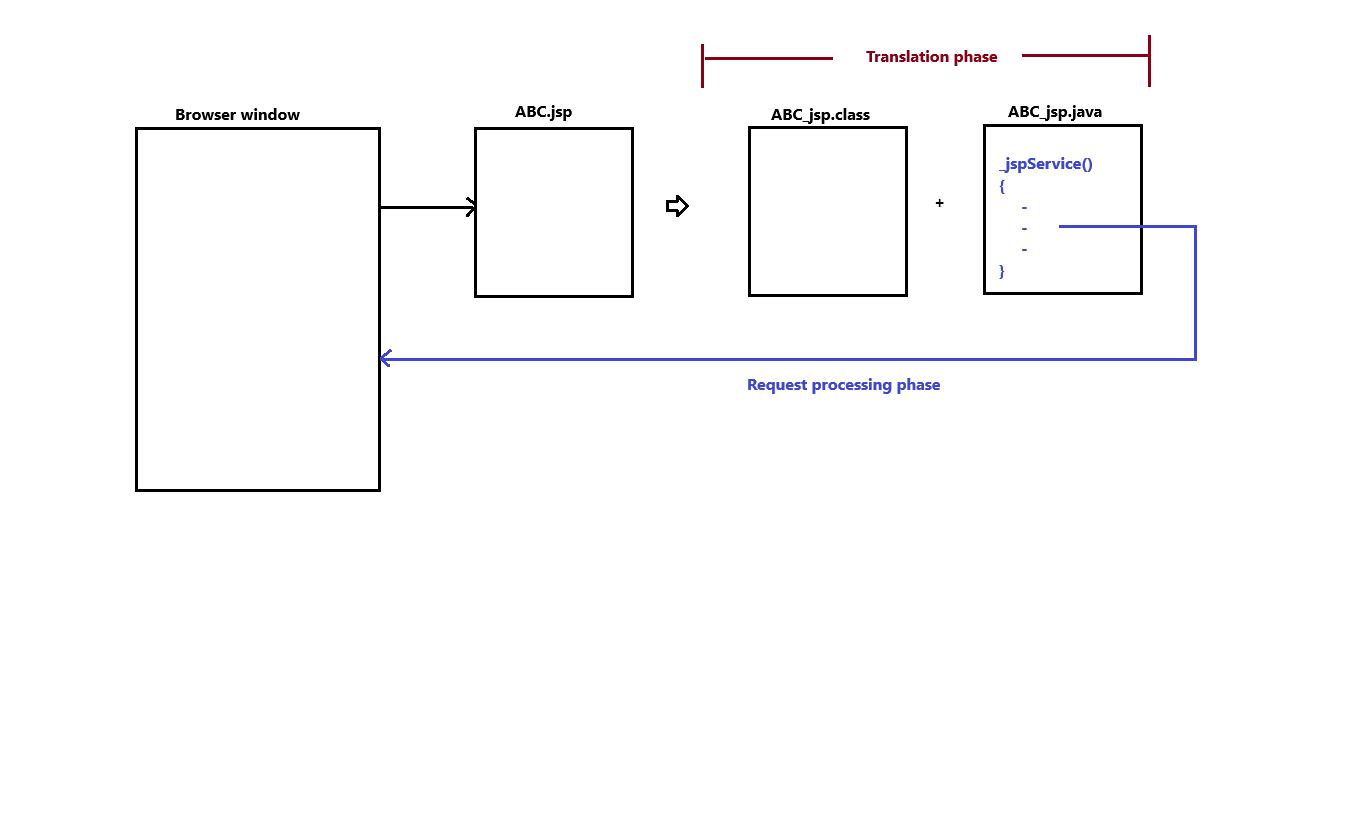
===============

We have two phases in JSP.

1) Translation phase

2) Request Processing phase

Diagram: jsp1.1



1) Translation phase

--------------------

In translation phase, our JSP program convert to JES class (ABC\_jsp.class & ABC\_jsp.java).

2) Request Processing phase

----------------------------

In request processing phase , our JES class will be executed and result send to browser window

as dynamic response.

How to enable <load-on-startup> and what happens if we enable <load-on-startup>

================================================================================

we can enable <load-on-startup> inside web.xml file.

web.xml

--------

<web-app>

<servlet>

<servlet-name>ABC</servlet-name>

<jsp-file>/WEB-INF/ABC.jsp</jsp-file>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>ABC</servlet-name>

<url-pattern>/test</url-pattern>

</servlet-mapping>

</web-app>

If we enable <load-on-startup> then our web container performs translation phase during the server startup or during the deployment of web application.

It means JES class object will be created before we give the first request.

Q) What is the difference between HTML and JSP?

HTML JSP

--------- --------

HTML stands for Hypertext Markup Language. JSP stands for Java Server Pages.

It is used to create static web pages. It is used to create dynamic web pages.

We can't create custom tags. We can create custom tags.

It requires browser window for execution. It requires server for execution.

It is used to build client side components. It is used to build server side components.

Interview Question

===================

Q) Write a java program to display the string in a given format?

input:

abc.txt

output:

txt

ex:

---

public class ExampleApp

{

public static void main(String[] args)

{

String fileName="abc.txt";

int index = fileName.lastIndexOf('.');

System.out.println(fileName.substring(index+1));

}

}

Class 2

Q) Types of errors in java?

We have three types of errors in java.

1) Logical Error

2) Compile time Error

3) Runtime Error

JSP Tags/Elements

==================

We have following three tags in JSP.

1) Scripting tags

----------------

i) Scriptlet tag

ex:

<% code here %>

ii) Expression tag

ex:

<%= code here %>

iii) Declaration tag

ex:

<%! code here %>

2) Directive tags

------------------

i) Page directive tag

ex:

<%@page attribute=value %>

ii) Include directive tag

ex:

<%@include attribute=value %>

3) Action Tags

----------------

<jsp:include>

<jsp:forward>

<jsp:useBean>

<jsp:setProperty>

<jsp:getProperty>

and etc.

Comments in JSP

-------------

<%-- comment here --%>

Scriptlet tag

=================

It is used to declare java code.

syntax:

<% code here %>

Deployment Directory Structure

------------------------------

JspApp2

|

|---Java Resources

|

|---WebContent

|

|---form.html

|---process.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

form.html

---------

<form action="process.jsp">

Name: <input type="text" name="t1"/>

<input type="submit" value="submit"/>

</form>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

process.jsp

------------

<%

String name = request.getParameter("t1");

out.println("Welcome :"+name);

%>

Request url

-----------

http://localhost:2525/JspApp2/

ii) Expression tag

====================

The code which is written in a expression tag will return to the output stream of a response.

Hence we don't need to use out.println() to print the data in a JSP.

syntax:

-------

<%= code here %>

Note:

Expression tag does not support semicolon.

form.html

---------

<form action="process.jsp">

Name: <input type="text" name="t1"/>

<input type="submit" value="submit"/>

</form>

process.jsp

-----------

<%

String name = request.getParameter("t1");

%>

<center>

<h1>

<%= "Hey! Welcome : "+name %>

</h1>

</center>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

Request url

---------

http://localhost:2525/JspApp2/

iii) Declaration tag

=====================

It is used to declare fields and methods.

syntax:

-------

<%! code here %>

Deployment Directory Structure

-----------------------------

JspApp3

|

|---Java Resources

|

|---WebContent

|

|---process.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

process.jsp

-----------

<%!

int data=100;

int cube(int n)

{

return n\*n\*n;

}

%>

<%= "The value is ="+data %> <br>

<%= "The cube of a given number is ="+cube(5) %>

web.xml

-----

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>process.jsp</welcome-file>

</welcome-file-list>

</web-app>

Request url

---------

http://localhost:2525/JspApp3/

Exception Handling in JSP

===========================

Exceptions are also known as runtime errors.

Exception may raise any time in our application so handling exceptions is always safer side for the programmer.

There are two ways to handle the exceptions in jsp.

1) Using errorPage and isErrorPage attribute of page directive tag.

2) Using <error-page> element in web.xml file.

1) Using errorPage and isErrorPage attribute of page directive tag

===================================================================

Deployment Directory Structure

-------------------------------

JspApp4

|

|---Java Resources

|

|---WebContent

|

|---form.html

|---process.jsp

|---error.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

form.html

----------

<form action="process.jsp">

No1: <input type="text" name="t1"/> <br>

No2: <input type="text" name="t2"/> <br>

<input type="submit" value="divide"/>

</form>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

process.jsp

------------

<%@page errorPage="error.jsp" %>

<%

String sno1=request.getParameter("t1");

String sno2=request.getParameter("t2");

int a=Integer.parseInt(sno1);

int b=Integer.parseInt(sno2);

int c = a / b;

%>

<center>

<h1>

<%= "Division of two numbers is ="+c %>

</h1>

</center>

error.jsp

---------

<%@page isErrorPage="true" %>

<b>

<i> Sorry! Exception occured!</i>

</b>

<br>

<%= exception %>

Request url

--------

http://localhost:2525/JspApp4/

2) Using <error-page> element in web.xml file

=============================================

This approach is better because we don't need to declare errorPage attribute in each jsp file.Defining <error-page> element as a single entry in web.xml file will handle all types of exceptions.

Deployment Directory Structure

-------------------------------

JspApp4

|

|---Java Resources

|

|---WebContent

|

|---form.html

|---process.jsp

|---error.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

form.html

----------

<form action="process.jsp">

No1: <input type="text" name="t1"/> <br>

No2: <input type="text" name="t2"/> <br>

<input type="submit" value="divide"/>

</form>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<error-page>

<exception-type>java.lang.Exception</exception-type>

<location>/error.jsp</location>

</error-page>

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

process.jsp

------------

<%

String sno1=request.getParameter("t1");

String sno2=request.getParameter("t2");

int a=Integer.parseInt(sno1);

int b=Integer.parseInt(sno2);

int c = a / b;

%>

<center>

<h1>

<%= "Division of two numbers is ="+c %>

</h1>

</center>

error.jsp

---------

<%@page isErrorPage="true" %>

<b>

<i> Sorry! Exception occured!</i>

</b>

<br>

<%= exception %>

Request url

--------

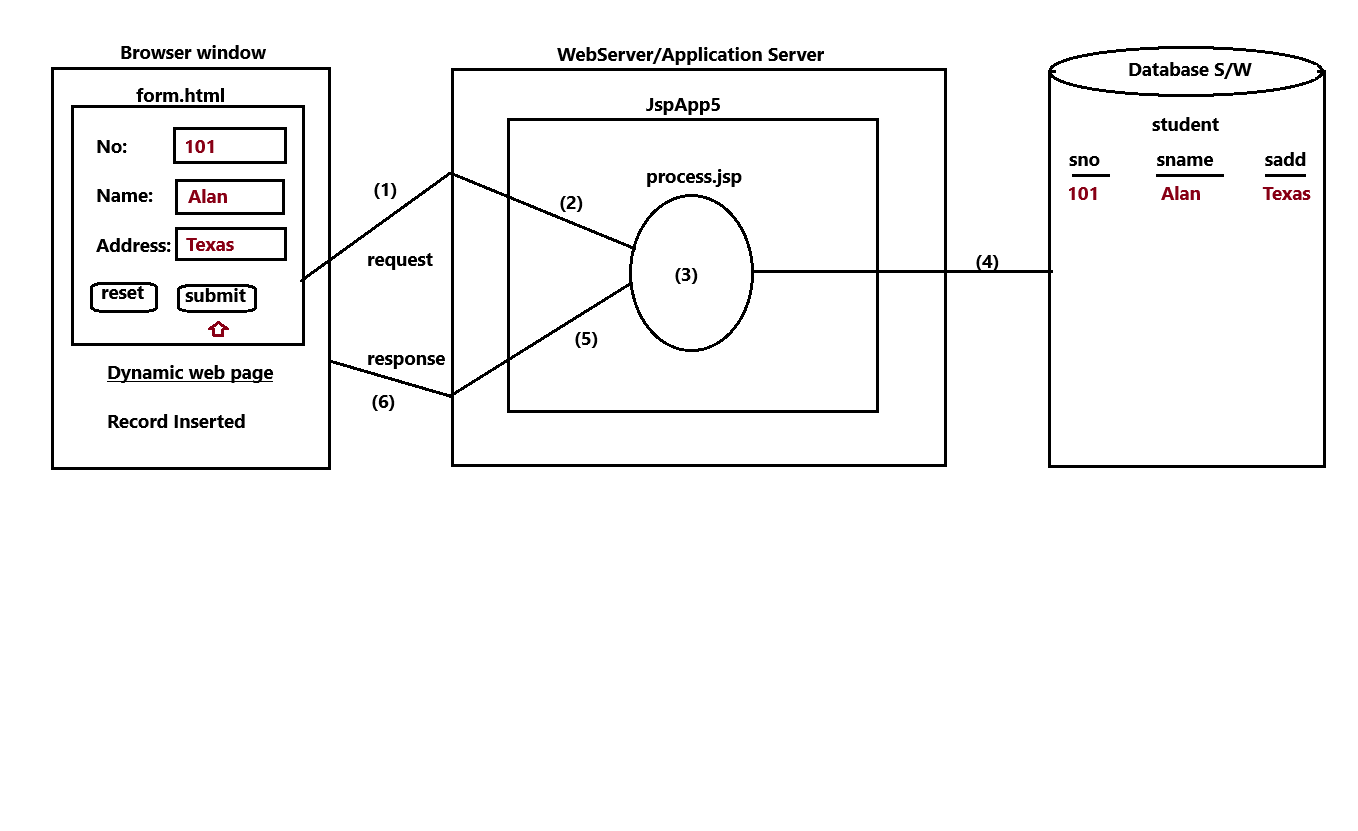
http://localhost:2525/JspApp4/

Class 3

JSP to Database Communication

==============================

Diagram: jsp3.1



student table

--------------

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12));

Deployment Directory Structure

------------------------------

JspApp5

|

|---Java Resources

|

|---WebContent

|

|---form.html

|---process.jsp

|

|---WEB-INF

|

|------web.xml

|

|------lib

|

|--ojdbc14.jar

Note:

-----

In above application we need to add "servlet-api.jar" and "ojdbc14.jar" file in project build path.

Copy and paste "ojdbc14.jar" file inside "WEB-INF/lib" folder seperately.

form.html

--------

<form action="process.jsp">

<table align="center">

<tr>

<td>No:</td>

<td><input type="text" name="t1"/></td>

</tr>

<tr>

<td>Name:</td>

<td><input type="text" name="t2"/></td>

</tr>

<tr>

<td>Address:</td>

<td><input type="text" name="t3"/></td>

</tr>

<tr>

<td><input type="reset" value="reset"/></td>

<td><input type="submit" value="submit"/></td>

</tr>

</table>

</form>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

process.jsp

------------

<%@page import="java.sql.\*" buffer="8kb" language="java"%>

<%

String sno=request.getParameter("t1");

int no=Integer.parseInt(sno);

String name=request.getParameter("t2");

String add=request.getParameter("t3");

//insert the data in a database

Connection con=null;

PreparedStatement ps=null;

int result=0;

String qry=null;

try

{

Class.forName("oracle.jdbc.driver.OracleDriver");

con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE","system","admin");

qry="insert into student values(?,?,?)";

ps=con.prepareStatement(qry);

//set the values

ps.setInt(1,no);

ps.setString(2,name);

ps.setString(3,add);

//excute

result=ps.executeUpdate();

if(result==0)

out.println("No Record Inserted");

else

out.println("Record Inserted");

ps.close();

con.close();

}

catch(Exception e)

{

out.println(e);

}

%>

Request url

----------

http://localhost:2525/JspApp5/

Action tags

============

Action tags are used to perform perticular task.

Action tags are used to control flow of pages and uses java beans.

Action tags are executed dynamically at runtime.

Action tags internally uses servlet API functionality.

Action tags are divided into two types.

1) Standard Action tags

2) Custom Action tags

1) Standard Action tags

------------------------

Built-In tags are called standard action tags.

ex:

<jsp:forward>

<jsp:include>

<jsp:useBean>

<jsp:setProperty>

<jsp:getProperty>

and etc.

Action forward

================

In action forward , the output of source JSP program will be discarded and output of destination JSP program goes to browser window as dynamic response.

It internally uses servlet API functionality called rd.forward(req,res).

syntax:

------

<jsp:forward page="page-name"/>

Deployment Directory Structure

------------------------------

JspApp6

|

|---Java Resources

|

|---WebContent

|

|---A.jsp

|---B.jsp

|

|---WEB-INF

|

|------web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

A.jsp

-----

<b><i>Begining of A.jsp file </i></b>

<br>

<jsp:forward page="B.jsp"/>

<br>

<b><i>Ending of A.jsp file </i></b>

B.jsp

------

<b><i> This is B.jsp file </i></b>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>A.jsp</welcome-file>

</welcome-file-list>

</web-app>

Request url

-------

http://localhost:2525/JspApp6/

Action include

==============

In action include, the output of source JSP program and output of destination JSP program combinely goes to browser window as dynamic response.

It internally uses servlet API functionality called rd.include(req,res).

syntax:

------

<jsp:include page="page\_name"/>

Deployment Directory Structure

------------------------------

JspApp6

|

|---Java Resources

|

|---WebContent

|

|---A.jsp

|---B.jsp

|

|---WEB-INF

|

|------web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

A.jsp

-----

<b><i>Begining of A.jsp file </i></b>

<br>

<jsp:include page="B.jsp"/>

<br>

<b><i>Ending of A.jsp file </i></b>

B.jsp

------

<b><i> This is B.jsp file </i></b>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>A.jsp</welcome-file>

</welcome-file-list>

</web-app>

Request url

-------

http://localhost:2525/JspApp6/

JSP to java bean communication

==============================

JSP to java bean communication is possible by using following three tags.

1) <jsp:useBean> tag

------------------

It is used to create and locate bean class object.

2) <jsp:setProperty> tag

-----------------------

It is used to set the values to bean object and calls setter methods.

3) <jsp:getProperty> tag

------------------------

It is used to get the values from bean object and calls getter methods.

Note:

------

All the above tags are independent tags.

Example1:

Deployment Directory structure

------------------------------

JspApp7

|

|---Java Resources

| |

|------src

|

|---com.ihub.www.

|

|---CuberNumber.java

|---WebContent

| |

|---process.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

process.jsp

------------

<jsp:useBean id="cn" class="com.ihub.www.CubeNumber"></jsp:useBean>

<center>

<h1>

<%= "Cube of a given number is ="+cn.cube(5) %>

</h1>

</center>

web.xml

--------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>process.jsp</welcome-file>

</welcome-file-list>

</web-app>

CubeNumber.java

--------------

package com.ihub.www;

public class CubeNumber

{

public int cube(int n)

{

return n\*n\*n;

}

}

Request url

---------

http://localhost:2525/JspApp7/

Assignment

==========

Q) Rohan is a kid who has just learned about creating words from alphabets. He has written some words in the notepad of his father's laptop. Now his father want to find the longest word written by Rohan using computer program. Write a program to find the longest string in a given list of Strings ?

input:

yes no number

output:

number

Clas 4

Example2:

Deployment Directory Structure

------------------------------

JspApp8

|

|---Java Resources

|

|-------src

|

|---com.ihub.www

|

|----User.java

|---WebContent

|

|---form.html

|---process.jsp

|

|---WEB-INF

|

|---web.xml

Note:

-----

In above application we need to add "servlet-api.jar" file in project build path.

form.html

----------

<form action="process.jsp">

UserName: <input type="text" name="username"/> <br>

Password: <input type="password" name="password"/> <br>

Email: <input type="email" name="email"/> <br>

<input type="submit" value="submit"/>

</form>

web.xml

---------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

User.java

-----------

package com.ihub.www;

public class User

{

private String username;

private String password;

private String email;

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

process.jsp

------------

<jsp:useBean id="u" class="com.ihub.www.User"></jsp:useBean>

<jsp:setProperty property="\*" name="u"/>

Records are <br>

<jsp:getProperty property="username" name="u"/> <br>

<jsp:getProperty property="password" name="u"/> <br>

<jsp:getProperty property="email" name="u"/> <br>

Request url

---------

http://localhost:2525/JspApp8/

Custom tags in JSP

==================

Tags which are created by the user based on the application requirements are called custom tags.

To create custom tags in JSP we need to taglib directive.

syntax:

-------

<%@taglib uri="uriofthetaglibrary" prefix="prefixoftaglibdirective" %>

To create a custom tag we need to configure tag information inside ".tld" file.

TLD stands for Tag Library Discriptor.

Deployment Directory Structure

--------------------------------

JspApp9

|

|---Java Resources

|

|------src

|

|---com.ihub.www

|

|----CubeNumber.java

|---WebContent

|

|---process.jsp

|

|---WEB-INF

|

|-----mytags.tld

|-----web.xml

|

|------lib

|

|---jsp-api.jar

Note:

------

In above application we need to add "servlet-api.jar" and "jsp-api.jar" file in project build path.

Copy and paste "jsp-api.jar" file inside "WEB-INF/lib" folder seperately.

process.jsp

-----------

<%@taglib uri="/WEB-INF/mytags.tld" prefix="ihub" %>

<center>

<h1>

<ihub:cube number="5"/>

</h1>

</center>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<welcome-file-list>

<welcome-file>process.jsp</welcome-file>

</welcome-file-list>

</web-app>

CubeNumber.java

--------------

package com.ihub.www;

import javax.servlet.jsp.JspException;

import javax.servlet.jsp.JspWriter;

import javax.servlet.jsp.tagext.TagSupport;

public class CubeNumber extends TagSupport

{

private int number;

//setter method

public void setNumber(int number)

{

this.number=number;

}

public int doStartTag()throws JspException

{

JspWriter out=pageContext.getOut();

try

{

out.println(number\*number\*number);

}

catch(Exception e)

{

e.printStackTrace();

}

return SKIP\_BODY;

}

}

mytags.tld

----------

<?xml version="1.0" encoding="ISO-8859-1" ?>

<!DOCTYPE taglib

PUBLIC "-//Sun Microsystems, Inc.//DTD JSP Tag Library 1.2//EN"

"http://java.sun.com/j2ee/dtd/web-jsptaglibrary\_1\_2.dtd">

<taglib>

<tlib-version>1.0</tlib-version>

<jsp-version>1.2</jsp-version>

<short-name>simple</short-name>

<uri>http://tomcat.apache.org/example-taglib</uri>

<tag>

<name>cube</name>

<tag-class>com.ihub.www.CubeNumber</tag-class>

<attribute>

<name>number</name>

<required>true</required>

</attribute>

</tag>

</taglib>

Request url

-----------

http://localhost:2525/JspApp9/

MVC in JSP

===========

MVC stands for Model View Controller.

MVC is a design pattern which seperates business logic, persistence logic and data.

Controller acts like a interface between model and view.

Controller is used to intercept all incoming request.

Model contains business logic and some times it is having data.

View is a presentation i.e GUI.

Diagram: jsp4.1

Deployment Directory Structure

--------------------------------

JspApp10

|

|---Java Resources

|

|------src

|

|---com.ihub.www

|

|----LoginSrv.java

|----LoginBean.java

|---WebContent

|

|---form.html

|---view.jsp

|---error.jsp

|

|---WEB-INF

|

|-----web.xml

Note:

------

In above application we need to add "servlet-api.jar" file in project build path.

form.html

-----------

<form action="test" method="GET">

<table align="center">

<tr>

<td>UserName:</td>

<td><input type="text" name="username"/></td>

</tr>

<tr>

<td>Password:</td>

<td><input type="password" name="password"/></td>

</tr>

<tr>

<td><input type="reset" value="reset"/></td>

<td><input type="submit" value="submit"/></td>

</tr>

</table>

</form>

web.xml

-------

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd" id="WebApp\_ID" version="3.0">

<servlet>

<servlet-name>LoginSrv</servlet-name>

<servlet-class>com.ihub.www.LoginSrv</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>LoginSrv</servlet-name>

<url-pattern>/test</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>form.html</welcome-file>

</welcome-file-list>

</web-app>

LoginBean.java

--------------

package com.ihub.www;

public class LoginBean

{

private String username;

private String password;

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

LoginSrv.java

--------------

package com.ihub.www;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class LoginSrv extends HttpServlet

{

protected void doGet(HttpServletRequest req,HttpServletResponse res)throws ServletException,IOException

{

PrintWriter pw=res.getWriter();

res.setContentType("text/html");

String uname=req.getParameter("username");

String pwd=req.getParameter("password");

//set the data to bean object

LoginBean lb=new LoginBean();

lb.setUsername(uname);

lb.setPassword(pwd);

//set the attribute to request object

req.setAttribute("bean", lb);

if(pwd.equals("admin"))

{

RequestDispatcher rd=req.getRequestDispatcher("view.jsp");

rd.forward(req, res);

}

else

{

RequestDispatcher rd=req.getRequestDispatcher("error.jsp");

rd.forward(req, res);

}

}

}

view.jsp

---------

<%@page import="com.ihub.www.LoginBean" %>

<%

LoginBean lb=(LoginBean)request.getAttribute("bean");

%>

<%= "UserName : "+ lb.getUsername() %> <br>

<%= "Password : "+ lb.getPassword() %> <br>

error.jsp

----------

<center>

<i>

<b style="color:red">Sorry! Incorrect username or password</b>

</i>

</center>

<%@include file="form.html" %>

Request url

----------

http://localhost:2525/JspApp10/