**Jsp’s [java server pages]**

**—----------**

By Using Java Programming Language we are able to prepare two types of applications.

1. Standalone Applications.

2. Enterprise Applications / Distributed Applications

Standalone Applications:

If we design and execute any application without using Client-Server arch or without distributing application logic over multiple machines then that application is called “Standalone Application”.

There are two types of Standalone Applications.

1. CUI Applications

2. GUI Applications

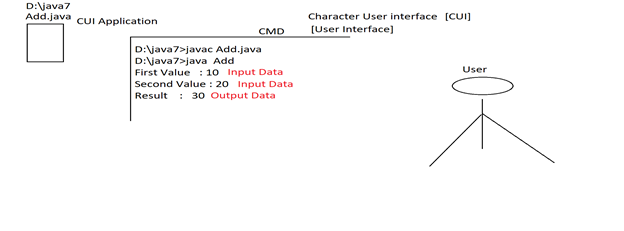
Q)What is the difference between CUI Application and GUI Application?

—-------------------------------------------------------------------------

Ans:

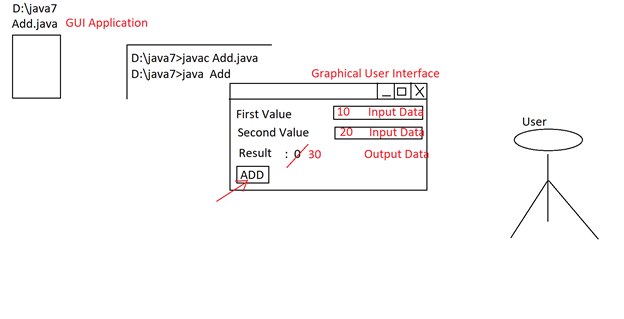
—---

CUI Application is a Standalone application , it was designed in such a way to take input data from users through the command prompt and to provide output data to the users through the same command prompt , here command prompt is acting as an user interface and it is supporting only characters data.



GUI application is a standalone application , It was designed in such a way to take input data through the collection of Graphic components and to provide output data through the same collection of Graphic Components, where the collection of Graphic components is called an User Interface and it is supporting Graphics data that is the data which is represented in the form of Graphic components .

To prepare Standalone Applications , JAVA has provided predefined Module in the form of J2SE, where J2SE has provided some predefined library in the form of the packages like java.lang, java.io, java.util, java.applet, java.awt, javax.swing,.......



Enterprise Applications or Distributed Applications:

If we design and execute any application on the basis of Client-Server arch or by distributing application logic over multiple machines then that application is called “Distributed application” or “Enterprise Applications”.

There are two types of Enterprise Applications.

1. Web Applications / Web based Distributed Applications

2. Distributed Applications / Remote Based Distributed Applications

**Q)What are the differences between Web applications and Distributed Applications?**

**—---------------------------------------------------------------**

**Ans:**

**—---**

**1.** **Web Application is a Client-Server application, where the complete application logic is provided at server machine.**

**Distributed application is a Client-Server application, where the complete application logic is distributed over multiple machines that are Local Machine and Remote machine.**

**2.** **In web applications, the Client is fixed, that is Browser.**

**In Distributed Applications, Client[Local Application] is not fixed, it may be a normal java program with main() method, it may be a GUI application with an applet, it may be a servlet , it may be a framework applications like Struts, JSF, Spring,.....**

**3.** **To prepare Web applications we have to use a set of technologies called “Web Technologies”.**

**EX: CGI,[common gateway interface] Servlets, JSPs, PHP, PERL, …….**

**To prepare Distributed Applications we have to use a set of technologies called “Distributed Technologies”.**

**EX: Socket Programming, RMI[remote method innvocation], CORBA[common object request broker architrcureagent], EJBs[enterprise java Beans], Web Services,.....**

**4.** **Web Applications are executed by both Web servers and Application Servers.**

**Distributed applications are executed by only Application Servers.**

**5.** **Web Application is the collection of Web components, where web components are executed by the web containers like Servlet container, JSP container,..... in the Servers.**

**Distributed Application is the collection of Distributed components like Java Beans, EJB components,..... Where the distributed components are executed by the Distributed Containers like EJB Container,...... in the Servers.**

**6.** **Web applications are mainly for static response and a little bit dynamic response from Servers.**

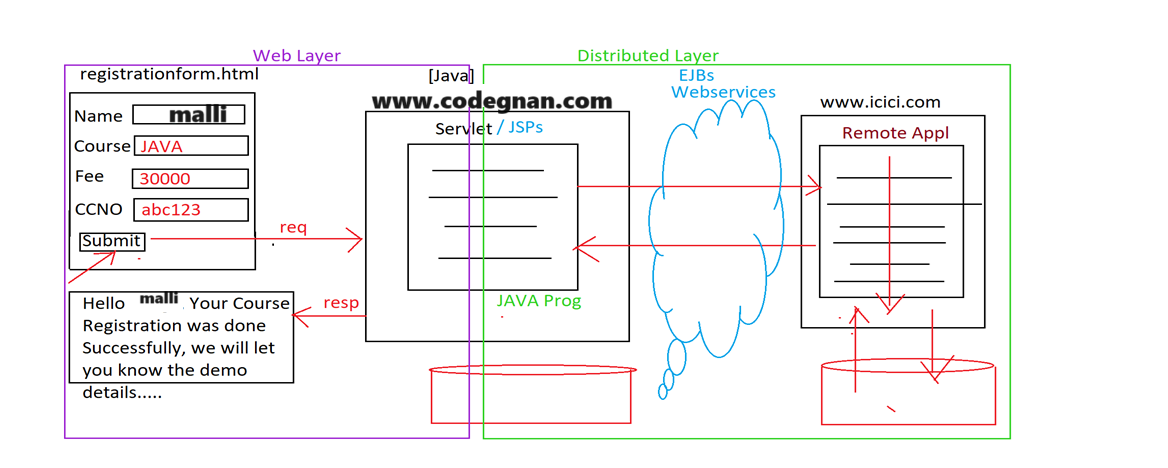
**Distributed applications are having main concentration on the**

**communication between Local machine and Remote Machine in**

**in order to consume remote services from Local machines.**

**In general, Enterprise Application is the combination of Web Layer and Distributed Application Layer, Where Web Layer is mainly for presentation in enterprise application and Distributed Application layer is mainly for Business logic.**

**Enterprise Application = Web Layer + Distributed Layer**

****

**In Web applications , there are two types of responses.**

**1.** **Static Response**

**2. Dynamic Response**

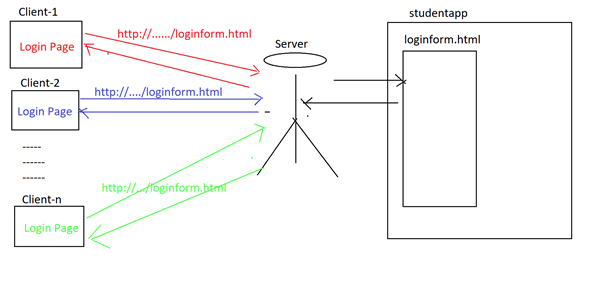
**Q)What is the difference between Static Response and Dynamic Response?**

**—------------------------------------------------------------------------------**

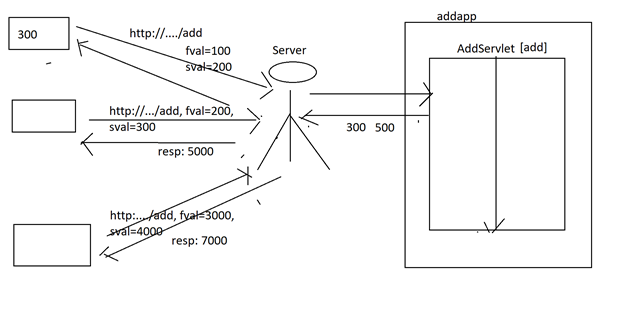
**Static Response is the response from Server machine , it will be generated without executing any resource at Server machine and it will be generated from server without performing any action at server side application.**

**Static response is the response which must be common to every user of that server side application.**

**EX: Every User will get a login form when we use** [**www.gmail.com**](http://www.gmail.com/)**.**

****

**Dynamic Response is the response from Server machine, it will be generated by executing a particular resource at server machine machine or by performing an action at server machine.**

****

**From the above representation, to generate dynamic response we have to use an application at server machine , that is web application, in web application we have to use some resources such as CGI, Servlets and JSPs.**

**From the above representation, the main purpose of “Servlets and JSPs” is to prepare web applications in order to generate “Dynamic Response”.**

**Q)To prepare web applications in order to generate dynamic response we have already CGI then what is the requirement to use Servlets?**

**—---------------------------------------------------------------------------**

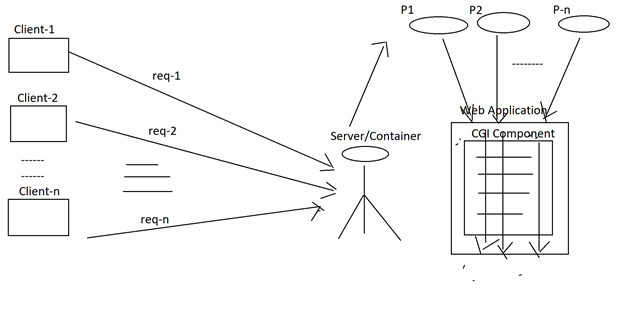
**Ans:**

**—---**

**CGI is a Server side technology, it can be used to prepare web applications , it was designed on the basis of C and some other scripting languages, it is a Process based technology, if we prepare web applications by using CGI then Container will create a separate process for each and every request coming from Client.**

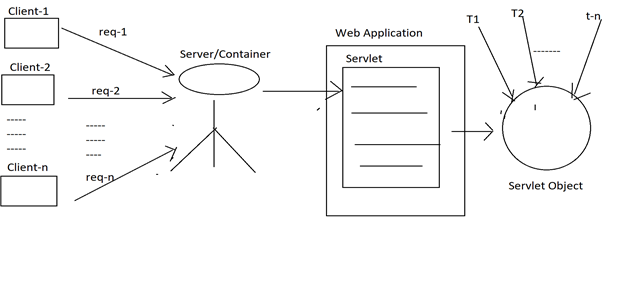
**Basically Process is heavy weight, so CGI is a heavy weight technology, it will take more memory and more execution time to process any request, it will take more response time, it will reduce server side application’s performance.**

**CGI is more error prone technology, more chances are available to get errors even in small pieces of code.**

****

**Servlets is a Server side technology to prepare web applications , it was designed on the top of Java, where JAVA is a thread based programming language, moreover Multi Threaded Language, so Servlets is also a thread based server side technology, if we design web applications by using servlets then Container will create a separate thread**

**instead of a process for each and every request coming from client, where Thread is a lightweight component, so Servlets is a lightweight Server side technology, it will take less memory and less execution time to process requests, it will reduce overall response time, it will increase web applications performance.**

****

**Q)To prepare web applications we already have Servlets then what is the requirement to use JSPs?**

**Or**

**Q)What are the differences between Servlets and JSPs?**

**—----------------------------------------------------------------------------**

**Ans:**

**—----**

**1.If we want to prepare web applications by using Servlets then Web application Developers must have very good knowledge on Java.**

**If we want to prepare web applications by using JSPs then web application developers are not required to have Java knowledge.**

**Note: The main intention of JSPs is to reduce Java code as much as possible in web applications.**

**2.In web applications , Servlets are very much suitable to take requests and to handle requests.**

**In web applications, JSPs are very much suitable for generating dynamic responses with a very good look and feel.**

**3.In web applications, Servlets are mainly for request processing.**

**In web applications, JSps are mainly for presentation.**

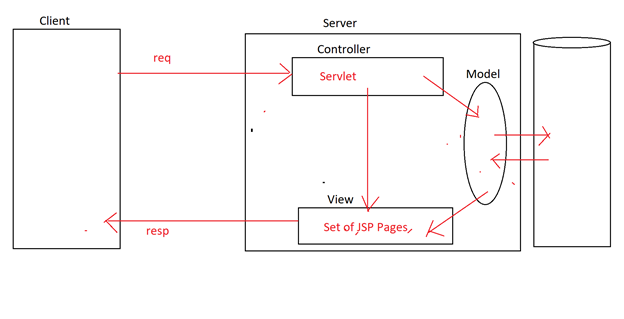
**4.In MVC based web applications, we will use a servlet as a controller.**

**IN MVC based web applications, we will use a set of JSP pages as the View part.**

**EX1: Struts is a MVC based web framework, it uses ActionServlet as a controller and a set of JSP pages as View part.**

**EX2: JSF is a MVC based web framework, it uses FacesServlet as controller and a set of JSP pages as View Part.**

**EX3: Spring WEB MVC module is a MVC based Web Framework, it uses DispatcherServlet as controller and a set of JSP pages as view part.**

****

**5.** **If we perform modifications on an existing Servlet in a web application then we must perform recompilation and reloading of that servlet in Server software.**

**If we perform modifications on an existing JSP page then it is not required to perform recompilation and reloading of that application into the server, because JSP pages are auto compiled and Auto Loaded.**

**6.** **If we prepare web applications by using only Servlets then it is very much difficult to separate presentation logic and Business logic.**

**If we prepare web applications by using JSPs then it is very**

**simple to separate presentation logic and business logic, because in JSP pages we will use Html tags to prepare presentation logic and we will use JSP tags to prepare business logic.**

**Java Server Pages:**

**—------------------**

**Java Server Pages is a Server side technology, it was designed on the basis of J2SE and Servlets API.**

**The main purpose of JSP technology is to reduce or avoid java code in web applications.**

**In web applications, JSP pages are executed by running Servlets internally.**

**If we submit a request to a particular JSP page , then the JSP Container will translate the JSP page to a Servlet internally and the Container will generate a response by executing the translated Servlet only.**

**In general,**

**1.** **We are able to prepare web applications by using only Servlets, but not suggestible.**

**2.** **We are able to prepare web applications by using only JSPs pages, but not suggestible.**

**3.** **It is suggested to use both Servlets and Jsp pages together to prepare web applications.**

**JSP is an abstraction provided by the SUN Microsystems and its implementations are provided by the Server vendors.**

**In general, in web applications, we are able to keep JSP pages under application folder[in Eclipse IDE, we will provide JSP pages under webapp folder], but it is possible to keep jsp pages at any location in the web application directory Structure like under application folder, Under WEB-INF folder and under classes folder..**

**If we keep JSP pages under the application folder that is in the public area then we are able to access that jsp file by using its name from the client, but if we provide JSP file under private area that is under WEB-INF or under classes folder then we are able to**

**EX:**

**Welcome.jsp**

**<html>**

**<body>**

**<h1 style="color: red;" align="center">**

**Welcome To First JSP Application**

**</h1>**

**</body>**

**</html>**

**hello.jsp**

**<html>**

**<body>**

**<h1 style="color: blue;" align="center">Hello User, This is from hello.jsp from WEB-INF folder.</h1>**

**</body>**

**</html>**

**hi.jsp**

**<html>**

**<body>**

**<h1 style="color: green;" align="center">Hi User, This is from hi.jsp from classes folder.</h1>**

**</body>**

**</html>**

**URLs:**

[**http://localhost:1010/jspapp01/welcome.jsp**](http://localhost:1010/jspapp01/welcome.jsp)

[**http://localhost:1010/jspapp01/helloJSP**](http://localhost:1010/jspapp01/helloJSP)

[**http://localhost:1010/jspapp01/hiJSP**](http://localhost:1010/jspapp01/hiJSP)

**JSP Elements:**

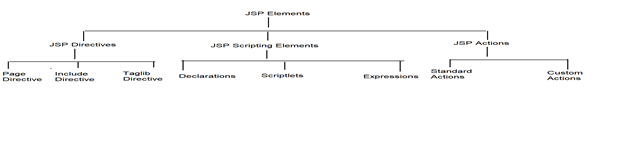
**—------------**

**To prepare JSP pages we have to use the following elements.**

**1.** **JSP Directives**

**2.** **Jsp Scriptting Elements**

**3.** **Jsp Actions**



**Q)In JSP pages, to prepare JSP pages already we have JSP Directives then what is the requirement to use Scripting elements?**

**Or**

**Q)What are the differences between JSP directives and Scripting elements?**

**—----------------------------------------------------------------------------**

**Ans:**

**—---**

**1. JSP directives are used to**

**a.** **Define current page characteristics.**

**b.** **Include target page content into the present JSP page.**

**c.** **To make available a user defined tag library in the present JSP page.**

**The main purpose of JSp scripting elements is to provide java code inside the JSP pages like below.**

**a.** **To declare variables, methods,... in the present JSp page.**

**b.** **To declare a block of java code to execute.**

**c.** **To represent and evaluate a Java expression.**

**2. Majority of the Directives will not give direct impact in the response generation.**

**Majority of the scripting elements will give a direct impact in the response generation.**

**3. Almost all the JSP directives are evaluated at the time of Translating JSp page to a Servlet, that is in Translation Phase**

**Almost all the JSP Scripting elements are evaluated at Request Processing time.**

**Q)To prepare JSP pages already we have JSP Scripting elements then what is the requirement to use JSP Actions?**

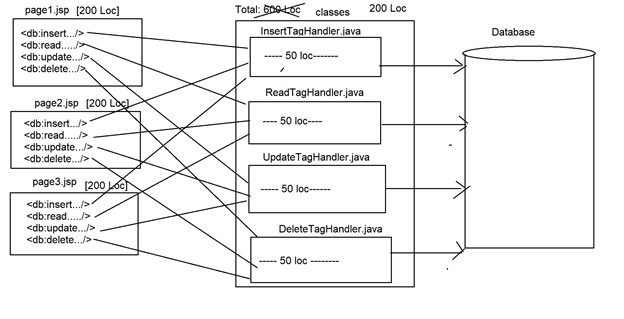
**—--------------------------------------------------------------------------**

**Ans:**

**—---**

**In JSP, the main purpose of JSP Scripting elements is to allow java code inside the JSP pages, but it is against to the JSP rules and regulations, here to avoid JAVA code in JSP pages we have to remove Scripting elements from JSP pages, to remove scripting elements from JSP pages we have to use an alternative , here the required alternative is “JSP Actions”.**

**In case of JSP Actions, we will define a tag in the JSP pages in place of Java code and we will provide the required java code under classes folder internally. In this context, When a JSP Container encounters the provided tag in the jsp page then Container will identify the internal Java code and Container will execute the internal Java code and perform a particular action, here the action which is performed through a tag is called a JSP action.**

****

**JSP Directives:**

**—---------------**

**The main purpose of JSP directives is**

**1.** **To define present page characteristics.**

**2.** **To include the target page content in the present jsp page.**

**3.** **To make available a User defined tag library in the present JSP page.**

**There are three types of JSP directives.**

1. **Page Directive**
2. **Include Directive**
3. **Taglib Directive.**

**Page Directive:**

**—---------------**

**The main purpose of page directive is to define present jsp page characteristics like**

1. **To define content type in the present JSP page.**
2. **To import the required java packages.**
3. **To define error pages**
4. **To allow or not to allow exception , session implicit objects**
5. **To define some description about the present JSP page**

**—-----**

**—------**

**There are two types of Syntaxes for the JSP Page Directive.**

1. **JSP Based Syntax.**

**<%@page [AttributesList] %>**

1. **XML Based Syntax.**

**<jsp:directive.page [AttributesList] />**

**Page Directive Attributes:**

1. **language**
2. **contentType**
3. **extends**
4. **import**
5. **info**
6. **errorPage**
7. **isErrorPage**
8. **buffer**
9. **autoFlush**
10. **session**
11. **isThreadThread**
12. **isELIgnored**

**language:**

**—----------**

**‘language’ attribute in the page directive is able to take a particular scripting language in order to allow the respective language code inside the scripting elements in jsp pages.**

**Almost all the Servers are supporting only JAVA as language to allow inside the Scripting elements.**

**The default value of this attribute is ‘java’.**

**EX: <%@page language=”java” %>**

**contentType:**

**—-----------**

**This attribute is able to take a particular MIME Type as a value in order to give an intimation to the respective client about to specify the type of response that present web resource is generating.**

**The default value for this attribute is “text/html”.**

**EX: <%@page contentType=”image/jpeg”%>**

**If we provide the value to the contentType attribute then the following actions will be performed.**

1. **The provided value will be set to the “ContentType” response header in the Response Format.**
2. **When Response Format reaches the client browser, Client Browser will take ContentType response header value that is the provided MIME Type.**
3. **On the basis of the ContentType response header value only Client browser will decide whether to display response in the browser to save response to a location in the Client System Hard Disk.**

**extends**

**—--------**

**This attribute can be used to provide a particular class name as value in order to set this class as a super class to the translated Servlet internally.**

**The default value for this attribute is “HttpJspBase”.**

**EX: <%@page extends = “MyClass”%>**

**Where MyClass must be the child class to HttpJspBase class.**

**import:**

**—--------**

**This attribute can be used to import java packages to the present JSPpage in order to use the classes and interfaces of the respective packages.**

**The default values of this attribute are “jakarta.servlet”, “jakarta.servlet.http”,.,...**

**EX:**

**<%@ page import=*"java.util.\*"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<h1>To Date Date : <%=new Date().toString() %></h1>**

**</body>**

**</html>**

**If we want to import more than one package into the present JSP page then we have to use the following cases.**

**Case#1: Use Multiple page directive.**

**<%@ page import=*"java.util.\*"*%>**

**<%@ page import=*"java.sql.\*"*%>**

**<%@ page import=*"java.io.\*"*%>**

**Case#2:Repeat import attribute multiple times in a single page directive.**

**EX:**

**<%@ page import=*"java.util.\*"* import=*"java.sql.\*"* import=*"java.io.\*"*%>**

**Case#3: Use a single import attribute and provide multiple packages with , separator.**

**<%@ page import=*"java.util.\*, java.sql.\*, java.io.\*"*%>**

**Note: In page directive only “import” attribute is repeatable attribute, no other attribute is repeatable.**

**info:**

**It can be used to provide some description about the present JSP page.**

**<%@ page info=”This is Welcome JSP” %>**

**If we provide description along with info attribute then we are able to get that description programmatically by using getServletInfo() method from Servlet interface.**

**The default value for this attribute is “Jasper JSP 3.0 Engine”.**

**EX:**

**<%@ page info=*"This is Test JSP Page"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<h1><%= getServletInfo() %></h1>**

**</body>**

**</html>**

**errorPage:**

**—---------**

**It can be used to take a particular error page name and location in order to forward requests to the error page if we have any exception in the present JSP page.**

**<%@ page errorPage=”error.jsp” %>**

**isErrorPage:**

**—-----------**

**It is a boolean attribute, it can be used to make the present JSP page as an error page, where error jsp page is a jsp page, it must have an implicit exception object.**

**This attribute can be used to send a request to the container about to allow or not to allow an exception implicit object in the present JSP page in order to display error messages.**

**If we provide true as a value to isErrorPage attribute then the container will allow exception implicit object in the present jsp page.**

**If we provide false as a value to the isErrorPage attribute then the container will not allow exception implicit object.**

**The default value for this attribute is “false”.**

**EX: <%@ page isErrorPage = true” %>**

**EX:**

**test.jsp**

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

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<%**

**java.util.Date d = null;**

**out.println(d.toString());**

**%>**

**</body>**

**</html>**

**error.jsp**

**<%@ page isErrorPage=*"true"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<h1 style="color: *red*;" align=*"center"*>**

**<%= exception %>**

**</h1>**

**</body>**

**</html>**

**Note: Use wildfly server for the above application.**

**session:**

**—-------**

**It is a boolean attribute, it can be used to make the container to allow or not to allow ‘session’ implicit object in the present JSP page.**

**If we provide true value to the ‘session’ attribute then the container is able to allow session implicit object in the present JSP page.**

**If we provide false value to the ‘session’ attribute then the container is unable to allow session implicit object in the present JSP page.**

**The default value of this attribute is ‘true’.**

**EX: <%@ page session=”false” %>**

**EX:**

**<%@ page session="true"%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="ISO-8859-1">**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<%**

**session.setAttribute("A", "AAA");**

**session.setAttribute("B", "BBB");**

**%>**

**<h1>**

**A-----><%= session.getAttribute("A") %><br>**

**B-----><%= session.getAttribute("B") %><br>**

**</h1>**

**</body>**

**</html>**

**Status: No Exception**

**OP:**

**A—----> AAA**

**B—----> BBB**

**EX:**

**<%@ page session="false"%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="ISO-8859-1">**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<%**

**session.setAttribute("A", "AAA");**

**session.setAttribute("B", "BBB");**

**%>**

**<h1>**

**A-----><%= session.getAttribute("A") %><br>**

**B-----><%= session.getAttribute("B") %><br>**

**</h1>**

**</body>**

**</html>**

**Status: Error: session can not be resolved.**

**Include directive:**

**—-----------------**

**The main purpose of the include directive is to include the target resource content into the present JSP page.**

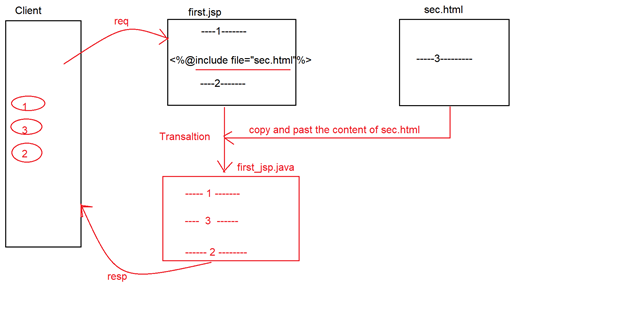
**Syntax:**

**JSP Based Syntax:**

**<%@ include file=”fileName” %>**

**XML Based Syntax:**

**<jsp:directive.include file=”fileName” %>**

****

**logo.jsp:**

**----------------------**

**<html>**

**<body><center>**

**<table width="100%" height="20%" bgcolor="red">**

**<tr><td colspan="2"><center><b><font size="7" color="white"> Newzen Software Solutions**

**</font></b></center></td></tr>**

**</table></center></body>**

**</html>**

**footer.jsp**

**----------------------------**

**<html>**

**<body><center>**

**<table width="100%" height="15%" bgcolor="blue">**

**<tr><td colspan="2"><center><b><font size="6" color="white"> copyrights2010-2020@newzensoftwaresolutions**

**</font></b></center></td></tr>**

**</table></center></body>**

**</html>**

**body.jsp**

**------------------------**

**<html>**

**<body bgcolor="lightyellow">**

**<center><b><font size="7">**

**<p><br>**

**newzeninfotech Solutions is one of the Training Institute.**

**<br><br></p>**

**</font></b></center></body>**

**</html>**

**mainpage.jsp**

**-------------------**

**<%@include file="logo.jsp"%>**

**<%@include file="body.jsp"%>**

**<%@include file="footer.jsp"%>**

**taglib directive:**

**—----------------**

**The main purpose of taglib directive is to make available user defined tags into the present JSP page.**

**Syntax:**

**JSP Based Syntax:**

**<%@taglib uri=”---” prefix=”--” %>**

**XML Based Syntax:**

**No XML based Syntax for Taglib directive.**

**Where uri attribute is able to take the name and location of the user defined tag library.**

**Where prefix attribute will define prefix value to the user defined tags.**

**EX:**

**<%@taglib uri=”db.tld” prefix=”db” %>**

**<db:select table=”emp1”/>**

**JSP Scripting Elements:**

**—-----------------------**

**The main purpose of Scripting elements is to provide java code inside the JSP pages.**

**There are three types of JSP Scripting elements.**

1. **Declarations**
2. **Scriptlets**
3. **Expressions**

**1. Declarations:**

**It is able to allow all java declarations like**

1. **Variables Declarations**
2. **Methods Declarations**
3. **Classes, abstract classes, interfaces declarations**

**—----**

**—----**

**Syntax:**

**<%!**

**—----Java declarations—----**

**%>**

**If we provide java declarations like variables, methods,.... Inside the Declarations scripting elements then the provided java declarations will be available as class level declarations in the translate servlet.**

**2. Scriptlets:**

**It is able to provide a block of Java code.**

**Syntax:**

**<%**

**—-----**

**—-Java Code—--**

**—------**

**%>**

**If we provide a block of java code by using scriptlets then the provided java code will be available inside the \_jspService() method in the translated servlet.**

**3. Expressions:**

**It is able to take a single java expression , evaluate it and display the result of the poression.**

**Syntax:**

**<%= JavaExpression %>**

**If we provide any java expression with JSP Expression scripting element then the provided java expression will be placed inside the \_jspService() method along with the out.write(JavaExpression);**

**Ex:**

**----------------**

**datetime.jsp**

**-------------------------**

**<%@ page import = "java.time.\*" %>**

**<%!**

**LocalDateTime dateTime = null;**

**String dateAndTime = "";**

**int day;**

**int month;**

**int year;**

**int hour;**

**int minute;**

**int second;**

**%>**

**<%**

**dateTime = LocalDateTime.now();**

**day = dateTime.getDayOfMonth();**

**month = dateTime.getMonthValue();**

**year = dateTime.getYear();**

**hour = dateTime.getHour();**

**minute = dateTime.getMinute();**

**second = dateTime.getSecond();**

**dateAndTime = day+"/"+month+"/"+year+" T"+hour+":"+minute+":"+second;**

**%>**

**<html>**

**<body>**

**<h1 style="color: red;" align="center">**

**Date and Time : <%= dateAndTime %>**

**</h1>**

**</body>**

**</html>**

**JSP Comments:**

**—-------------**

**In JSP pages , we are able to provide the following three types of comments.**

1. **XML Based Comments**
2. **JSP Based Comments**
3. **Java Based Comments**

**1. XML Based Comments:**

**<!--**

**—----Description—------**

**– ->**

**2. JSP Based Comments:**

**<%--**

**—----Description—----**

**– -%>**

**3. Java Based Comments:**

**These comments will be used in Scripting elements**

**There are three types of Java based comments.**

1. **Single line comment:**

**// —---Description—----**

1. **Multi line comment:**

**/\***

**—------**

**—------**

**\*/**

1. **Documentation Comment:**

**/\*\***

**\*------**

**\*------**

**—--**

**\*-----**

**\*/**

**Jsp implicit objects**

**—---------------------**

**In general, in web applications we will use some objects like request, response, session, exception,..... Frequently as per the application requirements, here to get these objects we have to use some java code explicitly, here to reduce java code we have to use JSP implicit objects.**

**JSP technology has provided the following implicit objects.**

**1.** **out —-----------> javax.servlet.jsp.JspWriter**

**2.** **request —-------> javax.servlet.http.HttpServletRequest**

**3.** **response —------> javax.servlet.http.HttpServletResponse**

**4.** **config —--------> javax.servlet.ServletConfig**

**5.** **application —---> javax.servlet.ServletContext**

**6.** **session —-------> javax.servlet.http.HttpSession**

**7.** **exception —-----> java.lang.Exception**

**8.** **page —----------> java.lang.Object**

**9.** **pageContext —---> javax.servlet.jsp.PageContext**

**Q)What is the difference between PrintWriter and JspWriter?**

**—-----------------------------------------------------------**

**Ans:**

**—---**

**PrintWriter is a Writer object in Servlets, it can be used to carry response , where PrintWriter is not a buffered writer, it will reduce the application performance.**

**JspWriter is a writer object in JSp technology, it can be used to carry responses from JSP pages, it is a Buffered Writer, it will improve application performance.**

**out: Represents the JspWriter object, used to send output to the client.**

**Example program:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<%**

**int num1 = 10;**

**int num2 = 20;**

**int num3 = num1 + num2;**

**out.println("num1 is " + num1 + "<br>");**

**out.println("num2 is " + num2 + "<br>");**

**out.println("The addition of num1 and num2 is " + num3 + "<br>");**

**%>**

**</body>**

**</html>**

**2.request**

* **The request object is an instance of java.servlet.http.HttpServletRequest and it is one of the argument of service method**
* **It will be created by container for every request.**
* **It will be used to request the information like parameter, header information , server name, etc.**
* **It uses getParameter() to access the request parameter.**

**Example:-**

**Index.jsp**

**------------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<form action=*"action.jsp"*>**

**<input type=*"text"* name=*"username"*>**

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

**</body>**

**</html>**

**action.jsp**

**----------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<% String userName=request.getParameter("username");**

**out.println("Welcome :" +userName);**

**%>**

**</body>**

**</html>**

**Response**

* **“Response” is an instance of class which implements HttpServletResponse interface**
* **Container generates this object and passes to \_jspservice() method as parameter**
* **“Response object” will be created by the container for each request.**
* **It represents the response that can be given to the client**
* **The response implicit object is used to content type, add cookie and redirect to response page**

**Example**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<%response.setContentType("text/html"); %>**

**</body>**

**</html>**

**Config**

* **“Config” is of the type java.servlet.servletConfig**
* **It is created by the container for each jsp page**
* **It is used to get the initialization parameter in web.xml**

**Example**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<% String servletName = config.getServletName();**

**out.println("Servlet Name is " +servletName);%>**

**</body>**

**</html>**

**Application**

* **Application object (code line 10) is an instance of javax.servlet.ServletContext and it is used to get the context information and attributes in JSP.**
* **Application object is created by container one per application, when the application gets deployed.**
* **Servletcontext object contains a set of methods which are used to interact with the servlet container.We can find information about the servlet container**

**Example:**

**--------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<% application.getContextPath(); %>**

**</body>**

**</html>**

**Session**

**-----------**

* **The session is holding “httpsession” object(code line 10).**
* **Session object is used to get, set and remove attributes to session scope and also used to get session information**

**Example**

**Index.jsp**

**--------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<% session.setAttribute("user","codegnan"); %>**

**<a href=*"action.jsp"*>Click here to get user name</a>**

**</body>**

**</html>**

**action.jsp**

**-------------------**

## **<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

## **pageEncoding=*"UTF-8"*%>**

## **<!DOCTYPE html>**

## **<html>**

## **<head>**

## **<meta charset=*"UTF-8"*>**

## **<title>Insert title here</title>**

## **</head>**

## **<body>**

## **<%**

## 

## **out.println(session.getAttribute("user"));**

## **%>**

## **</body>**

## **</html>**

## **PageContext**

* **This object is of the type of pagecontext.**
* **It is used to get, set and remove the attributes from a particular scope**

**JSP Scopes:**

**—-----------**

**In J2SE applications, to define scopes to the data we are able to use the access modifiers like private <default>, protected and public.**

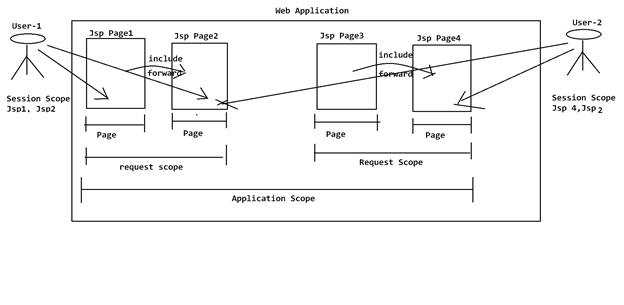
**Similarly, In JSP applications, to define scopes to the data we will use the following four types of Scopes.**

**1.** **Page Scope**

**2.** **Request Scope**

**3.** **Session Scope**

**4.** **Application Scope**

****

**Page Scope: if we declare data in the present jsp page then that data will have page scope and it will be available up to the present JSP page.**

**Request Scope: If we declare data in the request object then that data will have request scope and it will be available to all the JSP pages which are visited by the present request object.**

**Session Scope: If we declare data in HttpSession object then that data will have Session scope and it will be available to all the jsp pages which are accessed by the present User.**

**Application Scope: If we declare data in ServletContext object then that data will have Application Scope and it will be available to all the jsp pages throughout the application.**

**example**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<% pageContext.setAttribute("student","codegnan",pageContext.PAGE\_SCOPE);**

**String name = (String)pageContext.getAttribute("student");**

**out.println("student name is " +name);**

**%>**

**</body>**

**</html>**

**Code Line 11: we are setting the attribute using pageContext object, and it has three parameters:**

* **Key**
* **Value**
* **Scope**

**In the above code, the key is student and value is “gurustudent” while the scope is the page scope. Here the scope is “page” and it can get using page scope only.**

**Code Line 12: We are getting the value of the attribute using pageContext**

**When you execute the above code, you get the following output:**

**Output:**

* **The output will print “student name is gurustudent”.**

**Page:**

* **Page implicit variable holds the currently executed servlet object for the corresponding jsp.**
* **Acts as this object for current jsp page.**

**Example:**

**In this example, we are using page object to get the page name using toString method**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<% String pageName = page.toString();**

**out.println("Page Name is " +pageName);%>**

**</body>**

**</html>**

**Exception**

* **Exception is the implicit object of the throwable class.**
* **It is used for** [**exception handling in JSP**](https://www.guru99.com/jsp-exception-handling.html)**.**
* **The exception object can be only used in error pages.**

**Example:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"* isErrorPage=*"true"*%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert title here</title>**

**</head>**

**<body>**

**<%int[] num1={1,2,3,4};**

**out.println(num1[5]);%>**

**<%= exception %>**

**</body>**

**</html>**

**3. JSP ACTIONS:**

**In Jsp technology, by using scripting elements we are able to provide java code inside the Jsp pages, but the main theme of Jsp technology is not to allow java code inside Jsp pages.**

**To eliminate java code from Jsp pages we have to eliminate scripting elements, to eliminate scripting elements from Jsp pages we have to provide an alternative i.e. Jsp Actions.**

**In case of Jsp Actions, we will define a scripting tag in Jsp page and we will provide a block of java code w.r.t. scripting tag.**

**When container encounters the scripting tag then container will execute respective java code, by this an action will be performed called as Jsp Action.**

**In Jsp technology, there are 2 types of actions.**

**1. Standard Actions**

**2. Custom Actions**

**STANDARD ACTIONS:**

**Standard Actions are Jsp Actions, which could be defined by the Jsp technology to perform a particular action.**

**Jsp technology has provided all the standard actions in the form of a set of predefined tags called Action Tags.**

**• <jsp:useBean>**

**• <jsp:setProperty>**

**• <jsp:getProperty>**

**• <jsp:include >**

**• <jsp:forward >**

**• <jsp:param >**

**• <jsp:plugin >**

**• <jsp:fallback >**

**• <jsp:params >**

**• <jsp:declaration>**

**• <jsp:scriptlet >**

**• <jsp:expression >**

**JAVA BEANS:**

**Java Bean is a reusable component.**

**Java Bean is a normal java class which may declare properties, setter and getter methods in order to represent a particular user form at server side.**

**If we want to prepare Java Bean components then we have to use the following rules and regulations.**

**1. Java Bean is a normal java class, it is suggestible to implement Serializable interface.**

**2. Always Java Bean classes should be public, non-abstract and non-final.**

**3. In Java Bean classes, we have to declare all the properties w.r.t. the properties define in the respective user form.**

**4. In Java Bean classes, all the properties should be private.**

**5. In Java Bean classes, all the behaviours should be public.**

**6. If we want to declare any constructor in Java Bean class then that constructor should be public and zero argument.**

**EX:**

**public class Employee implements Serializable**

**{**

**private String eno;**

**private String ename;**

**private float esal;**

**public void setEno(String eno)**

**{**

**this.eno=eno;**

**}**

**public void setEname(String ename)**

**{**

**this.ename=ename;**

**}**

**public void setEsal(String esal)**

**{**

**this.esal=esal;**

**}**

**public String getEno()**

**{**

**return eno;**

**}**

**public String getEname()**

**{**

**return ename;**

**}**

**public float getEsal()**

**{**

**return esal;**

**}**

**}**

**1. <JSP:USEBEAN>:**

**The main purpose of <jsp:useBean> tag is to interact with bean object from a particular Jsp page. SYNTAX: <jsp:useBean id=”--” class=”--” type=”--” scope=”--”/>**

**Where id attribute will take a variable to manage generated Bean object reference. Where class attribute will take the fully qualified name of Bean class.**

**Where type attribute will take the fully qualified name of Bean class to define the type of variable in order to manage Bean object reference.**

**Where scope attribute will take either of the Jsp scopes to Bean object.**

**NOTE: In <jsp:useBean> tag, always it is suggestible to provide either application or session scope to the scope attribute value.**

**EX: <jsp:useBean id=”e” class=”Employee” type=”Employee” scope=”session”/>**

**When container encounters the above tag then container will pick up class attribute value i.e. fully qualified name of Bean class then container will recognize Bean class .class file and perform Bean class loading and instantiation.**

**After creating Bean object container will assign Bean object reference to the variable specified as value to id attribute.**

**After getting Bean object reference container will store Bean object in a scope specified as value to scope attribute.**

**2. <JSP:SETPROPERTY>:**

**The main purpose of <jsp:setProperty> tag is to execute a particular setter method in order to set a value to a particular Bean property.**

**SYNTAX: <jsp:setProperty name=”--” property=”--” value=”--”/>**

**Where name attribute will take a variable which is same as id attribute value in <jsp:useBean> tag. Where property attribute will take a property name in order to access the respective setter method. Where value attribute will take a value to pass as a parameter to the respective setter method.**

**3. <JSP:GETPROPERTY>:**

**The main purpose of <jsp:getProperty> tag is to execute a getter method in order to get a value from Bean object.**

**SYNTAX: <jsp:getProperty name=”--” property=”--”/>**

**Where name attribute will take a variable which is same as id attribute value in <jsp:useBean> tag. Where property attribute will take a particular property to execute the respective getter method**

**index.html**

**--------------------**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Person Registration</title>**

**</head>**

**<body>**

**<h2>Person Registration Form</h2>**

**<form method=*"post"* action=*"display.jsp"*>**

**<label for=*"name"*>Name:</label>**

**<input type=*"text"* name=*"name"* required>**

**<br>**

**<label for=*"age"*>Age:</label>**

**<input type=*"number"* name=*"age"* required>**

**<br>**

**<input type=*"submit"* value=*"Submit"*>**

**</form>**

**</body>**

**</html>**

**Person.java**

**---------------------------**

**package com.codegnan.beans;**

**public class Person {**

**private String name;**

**private int age;**

**public String getName() {**

**return name;**

**}**

**public void setName(String name) {**

**this.name = name;**

**}**

**public int getAge() {**

**return age;**

**}**

**public void setAge(int age) {**

**this.age = age;**

**}**

**}**

**display.jsp**

**------------------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"* import=*"com.codegnan.beans.Person"*%>**

**<%**

**// Capture form parameters**

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

**int age = Integer.parseInt(request.getParameter("age"));**

**%>**

**<jsp:useBean id=*"person"* class=*"com.codegnan.beans.Person"***

**scope=*"session"*>**

**<jsp:setProperty name=*"person"* property=*"name"* value=*"*<%=name%>*"* />**

**<jsp:setProperty name=*"person"* property=*"age"* value=*"*<%=age%>*"* />**

**</jsp:useBean>**

**<html>**

**<head>**

**<title>Display Person Information</title>**

**</head>**

**<body>**

**<h2>Person Information</h2>**

**<p>**

**Name:**

**<jsp:getProperty name=*"person"* property=*"name"* /></p>**

**<p>**

**Age:**

**<jsp:getProperty name=*"person"* property=*"age"* /></p>**

**</body>**

**</html>**

**4. <jsp:include>:**

**—-----------------**

**The main purpose of <jsp:include> tag is to include the target resource response in the present JSP page response.**

**This action tag is the same as the include request dispatching mechanism.**

**Syntax:**

**<jsp:include page=”---” flush=”--”/>**

**Where “page” attribute is able to take target resource name and location.**

**Where “flush” attribute is a boolean attribute, it is able to make the container either to Flush the response to the client when JspWriter’s Buffer is full with the response or to raise an exception when JspWriter’s buffer while including target resource response.**

**EX:**

**<jsp:include page=”loginform.html” flush=”true”/>**

**1. Jsp include example**

**Index.jsp:**

**-----------**

**<%@ page contentType="text/html; charset=UTF-8" language="java" %>**

**<html>**

**<head>**

**<title>Index Page</title>**

**</head>**

**<body style="background-color: powderblue;">**

**<form method="post" action="processInput.jsp">**

**<h3 style="text-align: center;">Enter a Number:</h3>**

**<div style="text-align: center;">**

**<input type="text" name="userInput">**

**<input type="submit" value="Submit">**

**</div>**

**</form>**

**</body>**

**</html>**

**processInput**

**<%@ page contentType=*"text/html; charset=UTF-8"* language=*"java"* %>**

**<html>**

**<head>**

**<title>Greet Page</title>**

**</head>**

**<body style="background-color: *powderblue*;">**

**<%**

**String userInput = request.getParameter("userInput");**

**Integer inputInteger = Integer.valueOf(userInput);**

**int updatedValue = inputInteger.intValue() + 50;**

**Integer updatedInteger = Integer.valueOf(updatedValue);**

**request.setAttribute("updatedValue", updatedInteger);**

**%>**

**<jsp:include page=*"updateValue.jsp"* />**

**<h3 style="text-align: *center*;">The Final updated Number is: <%= request.getAttribute("finalValue") %> </h3>**

**</body>**

**</html>**

**updateValue.jsp**

**--------------------**

**<%@ page contentType=*"text/html; charset=UTF-8"* language=*"java"* %>**

**<%**

**Integer updatedValue = (Integer)request.getAttribute("updatedValue");**

**int updatedIntValue = updatedValue.intValue();**

**updatedIntValue += 100;**

**Integer finalUpdatedValue = Integer.valueOf(updatedIntValue);**

**request.setAttribute("finalValue", finalUpdatedValue);**

**%>**

**<jsp:forward>**

**—---------------**

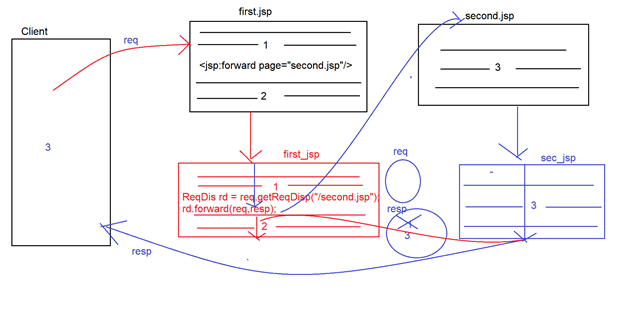
**The main purpose of the <jsp:forward> action tag is to forward requests from the present jsp page to the target resource.**

**Syntax:**

**<jsp:forward page=”--”/>**

**Where the page attribute is able to take the name and location of the target resource.**

**Note: It is the same as the Forward request dispatching mechanism.**

****

**2.jsp forward action example:**

**Show.jsp**

**-----------------------**

**<%@ page contentType=*"text/html; charset=UTF-8"* language=*"java"* %>**

**<html>**

**<head>**

**<title>Input Page</title>**

**</head>**

**<body bgcolor=*"powderblue"*>**

**<form method=*"post"* action=*"processInput.jsp"*>**

**<h3 style="text-align:*center*;">**

**<label for=*"text"*>Enter a Number:</label>**

**<input type=*"text"* name=*"userInput"*>**

**<input type=*"submit"* value=*"Submit"*>**

**</h3>**

**</form>**

**</body>**

**</html>**

**ProcessInput.jsp**

**--------------------**

**<%@ page contentType=*"text/html; charset=UTF-8"* language=*"java"* %>**

**<html>**

**<body bgcolor=*"pink"*>**

**<%**

**String userInput = request.getParameter("userInput");**

**Integer intValue = Integer.valueOf(userInput);**

**int updatedValue = intValue.intValue() + 50;**

**Integer updatedInteger = Integer.valueOf(updatedValue);**

**request.setAttribute("updatedValue", updatedInteger);**

**%>**

**<jsp:forward page=*"DisplayResult.jsp"*/>**

**</body>**

**</html>**

**DisplayResult.jsp**

**-------------------**

**<%@ page contentType=*"text/html; charset=UTF-8"* language=*"java"* %>**

**<html>**

**<body bgcolor=*"powderblue"*>**

**<h3 style="text-align:*center*;">**

**<%**

**Integer finalValue = (Integer)request.getAttribute("updatedValue");**

**int finalIntValue = finalValue.intValue() + 100;**

**Integer finalUpdatedValue = Integer.valueOf(finalIntValue);**

**out.println("The final updated number is: " + finalUpdatedValue);**

**%>**

**</h3>**

**</body>**

**</html>**

**<jsp:param> tag:**

**—----------------**

**It will be used as a child tag to <jsp:include> and <jsp:forward> , it can be used to provide a parameter to the request object at the time of forwarding request to the target page in <jsp:include> tag and in <jsp:forward> tag.**

**Syntax:**

**<jsp:param name=”--” value=”--”/>**

**EX:**

**loginform.jsp**

**-----------------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* language=*"java"*%>**

**<html>**

**<head>**

**<title>Login Form</title>**

**</head>**

**<body>**

**<form method=*"post"* action=*"./login.jsp"*>**

**<table align=*"center"*>**

**<tr>**

**<td>User Name</td>**

**<td><input type=*"text"* name=*"uname"*></td>**

**</tr>**

**<tr>**

**<td>User Password</td>**

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

**</tr>**

**<tr>**

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

**</tr>**

**</table>**

**</form>**

**</body>**

**</html>**

**login.jsp**

***--------------------------***

**<%@ page contentType=*"text/html;charset=UTF-8"* language=*"java"*%>**

**<%!String uname;**

**String upwd;**

**String status;%>**

**<%**

**uname = request.getParameter("uname");**

**upwd = request.getParameter("upwd");**

**if (uname.equals("malli") && upwd.equals("malli")) {**

**status = "User Login Success";**

**} else {**

**status = "User Login Failure";**

**}**

**%>**

**<jsp:forward page=*"status.jsp"*>**

**<jsp:param name=*"status"* value=*'*<%=status%>*'* />**

**</jsp:forward>**

**status.jsp**

**------------------------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* language=*"java"*%>**

**<html>**

**<head>**

**<title>Title</title>**

**</head>**

**<body>**

**<h1 style="color: *red*;" align=*"center"*>**

**Status :**

**<%=request.getParameter("status")%>**

**</h1>**

**<hr>**

**</body>**

**</html>**

**<jsp:include page=*"loginform.jsp"* flush=*"true"* />**

**<jsp:plugin>:**

**—-------------**

**The main purpose of this tag is to include an applet into the present JSP Page.**

**<jsp:plugin code=”---” width=”--” height=”--”>**

**—----**

**</jsp:plugin>**

**Where “code” attribute is able to take the name and location of the Applet class.**

**Where “width” and “height” attributes are able to take applet size.**

**EX:**

**<jsp:plugin code=”com.mallisoft.applets.LogoApplet” width=”500” height=”500”>**

**—-----**

**</jsp:plugin>**

**<jsp:fallback>:**

**—----------------**

**This tag will be used as a child tag to <jsp:plugin>**

**It can be used to display an alternative message when the specified applet is not initialized.**

**Syntax:**

**<jsp:plugin …..… >**

**<jsp:fallback>**

**—----Description—------**

**</jsp:fallback>**

**</jsp:plugin>**

**<jsp:params>**

**—-------------**

**It will be used as a child tag to <jsp:plugin> tag.**

**It can be used to provide some input parameters to the Applet in order to perform a particular action.**

**Syntax:**

**<jsp:params>**

**<jsp:param name=”--” value=”--”/**

**—----**

**—-----**

**</jsp:params>**

**<jsp:declaration>:**

**—------------------**

**It is the same as the scripting element declaration.**

**It is able to include all java declarations like variables, methods,....**

**Syntax:**

**<jsp:declaration>**

**—--Java Declarations—--**

**</jsp:declaration>**

**<jsp:scriptlet>:**

**—-----------------**

**It is the same as the Scripting element Scriptlet.**

**It is able to include a block of java code in order to execute.**

**Syntax:**

**<jsp:scriptlet>**

**—--A block of Java Code—---**

**</jsp:scriptlet>**

**<jsp:expression>:**

**—------------------**

**It is the same as the Scripting element expression.**

**It is able to represent, evaluate and display the result of a java expression.**

**Syntax:**

**<jsp:expression>**

**—-Java Expression—-**

**</jsp:expression>**

**EX:**

**date.jsp**

**------------------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* language=*"java"***

**import=*"java.time.\*"*%>**

**<jsp:declaration>LocalDateTime dateTime;**

**String val;</jsp:declaration>**

**<jsp:scriptlet>dateTime = LocalDateTime.now();**

**val = dateTime.toString();</jsp:scriptlet>**

**<html>**

**<body>**

**<h1 style="color: *red*;" align=*"center"*>**

**To Day Date and Time :**

**<jsp:expression>val</jsp:expression>**

**</h1>**

**</body>**

**</html>**

**JSTL [JSP Standard Tag Library]**

**—--------------------------------**

**The main purpose of Scripting elements in JSP is to provide java code in Jsp pages, it is against the JSp rules and regulations, here to preserve JSP rules and regulations we have to remove java code from JSp pages, for this we have to use JSP actions.**

**JSTL has provided the following types tags.**

**1.** **Core Tags**

**2.** **XML Tags**

**3.** **I18n Tags / Formatted Tags**

**4.** **SQL Tags**

**5.** **Functions Tags**

**To get all the above tag libraries into the present JSP page we have to use the following URLs to the uri attribute in taglib directive.**

**Core tags :**  [**http://java.sun.com/jstl/core**](http://java.sun.com/jstl/core)

**XML tags :**  [**http://java.sun.com/jstl/xml**](http://java.sun.com/jstl/xml)

**Formatted Tags :**  [**http://java.sun.com/jstl/fmt**](http://java.sun.com/jstl/fmt)

**SQL tags :** [**http://java.sun.com/jstl/sql**](http://java.sun.com/jstl/sql)

**Functions tags :** [**http://java.sun.com/jsp/jstl/functions**](http://java.sun.com/jsp/jstl/functions)

**Core Tags:**

**—----------**

**There are four types of Core tags.**

**1.** **General Purpose Tags**

**a.** **<c:set>**

**b.** **<c:remove>**

**c.** **<c:catch>**

**d.** **<c:out>**

**2. Conditional Tags**

**a.** **<c:if>**

**b.** **<c:choose>**

**c.** **<c:when>**

**d.** **<c:otherwise>**

**3. Iterative tags**

**a.** **<c:forEach>**

**b.** **<c:forTokens>**

**4. URL Based tags**

**a.** **<c:import>**

**b.** **<c:url>**

**c.** **<c:redirect>**

**<c:out>:**

**—--------**

**It can be used to display data on the browsers, it is the same as out.println() in Servlets.**

**<c:out value=”---”/>**

**If we want to display the value of a variable then we have to use the following expression as a value attribute.**

**${varName}**

**<c:out value=”${a}”/>**

**<c:set>**

**—-------**

**It can be used to set a variable and its data in a particular scope.**

**<c:set var=”--” value=”--” scope=”--”/>**

**Where the var attribute is able to take variable names.**

**Where the value attribute will take the value of the variable.**

**Where the scope attribute will take a particular scope like page, request,session and application to store data.**

**<c:remove>**

**—-----------**

**It can be used to remove data from the specified scope.**

**<c:remove var=”--” scope=”--”/>**

**Where “var” attribute will take a variable name which we want to remove.**

**Where “scope” attribute will take a particular scope to remove data.**

**Index.jsp**

**=========**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>Codegnan IT Solutions</title>**

**</head>**

**<body>**

**<h1>**

**<c:out value=*"Codegnan IT Solutions"* />**

**<br>**

**<%-- Use descriptive variable names --%>**

**<c:set var=*"sessionVariable"* value=*"AAA"* scope=*"session"* />**

**<c:set var=*"applicationVariable"* value=*"BBB"* scope=*"application"* />**

**Session Variable:**

**<c:out value=*"*${sessionVariable}*"* />**

**<br> Application Variable:**

**<c:out value=*"*${applicationVariable}*"* />**

**<br>**

**<%-- Remove the session variable --%>**

**<c:remove var=*"sessionVariable"* scope=*"session"* />**

**Session Variable after removal:**

**<c:out value=*"*${sessionVariable}*"***

**default=*"Variable 'sessionVariable' is removed"* />**

**</h1>**

**</body>**

**</html>**

**c:catch>**

**—-------**

**It can be used to catch an exception and it is able to display the related exception details.**

**<c:catch var=”--”>**

**—----**

**</c:catch>**

**Where var attribute is able to take a variable to refer the generated exception.**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<%**

**try {**

**int result = 100 / 0;**

**pageContext.setAttribute("result", result);**

**} catch (ArithmeticException e) {**

**pageContext.setAttribute("error", e.getMessage());**

**}**

**%>**

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>Error Handling</title>**

**</head>**

**<body>**

**<h1>**

**<c:choose>**

**<c:when test=*"*${not empty error}*"*>**

**Error: <c:out value=*"*${error}*"* />**

**</c:when>**

**<c:otherwise>**

**Result: <c:out value=*"*${result}*"* />**

**</c:otherwise>**

**</c:choose>**

**</h1>**

**</body>**

**</html>**

**<c:if>**

**—------**

**It will represent an ‘if’ conditional statement in Java.**

**<c:if test=”--”/>**

**Ex:**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<%**

**// Avoid using scriptlets for variable initialization.**

**// Instead, initialize variables using JSTL.**

**// For demonstration purposes, assuming 'a' is a dynamic value retrieved from the server.**

**int a = 10;**

**pageContext.setAttribute("a", a);**

**%>**

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>JSP If Statement</title>**

**</head>**

**<body>**

**<h1>**

**<%-- Use JSTL <c:set> to initialize variables --%>**

**<c:set var=*"a"* value=*"10"* />**

**<%-- Use <c:if> to conditionally display content --%>**

**<c:if test=*"*${a != 10}*"*>**

**<h1>a value is 10</h1>**

**</c:if>**

**<h1>After if</h1>**

**</h1>**

**</body>**

**</html>**

**<c:choose>, <c:when>, <c:otherwise>**

**—-----------------------------------**

**These tags are representing switch conditional statements in Java.**

**<c:choose> —-----> switch**

**<c:when test=”--”> —---------> case**

**—-----**

**</c:when>**

**<c:when test=”--”>**

**—-----**

**</c:when>**

**—------**

**<c:otherwise> —---------------> default**

**—---**

**</c:otherwise>**

**</c:choose>**

**EX:**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<%**

**int a = 20;**

**pageContext.setAttribute("a", a);**

**%>**

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>JSP Choose Statement</title>**

**</head>**

**<body>**

**<h1>**

**<%-- Use JSTL <c:set> to initialize variables --%>**

**<c:set var=*"a"* value=*"20"* />**

**<%-- Use <c:choose> to select from multiple conditions --%>**

**<c:choose>**

**<c:when test=*"*${a==5}*"*>**

**<h1>Five</h1>**

**</c:when>**

**<c:when test=*"*${a==10}*"*>**

**<h1>Ten</h1>**

**</c:when>**

**<c:when test=*"*${a==15}*"*>**

**<h1>Fifteen</h1>**

**</c:when>**

**<c:when test=*"*${a==20}*"*>**

**<h1>Twenty</h1>**

**</c:when>**

**<c:otherwise>**

**<h1>Please provide the Number in 5,10,15, or 20</h1>**

**</c:otherwise>**

**</c:choose>**

**</h1>**

**</body>**

**</html>**

**<c:foreach>**

**—-----------**

**It is same as forEach loop in Java, it is able to provide iterations over the body while reading elements from the specified Array or from the specified Collection.**

**<c:forEach var=”--” begin=”--” end=”--” step=”--”>**

**—-----**

**</c:forEach>**

**Where “var” attribute is able to take a variable to loop loop index value at each and every iteration.**

**Where the “begin” attribute will take the start index value.**

**Where the “end” attribute will take the end index value.**

**Where “step” will take increment value.**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>JSP forEach Loop</title>**

**</head>**

**<body>**

**<%-- Use JSTL forEach loop to iterate from 0 to 10 --%>**

**<c:forEach var=*"a"* begin=*"0"* end=*"10"*>**

**<c:out value=*"*${a}*"* />**

**<br>**

**</c:forEach>**

**</body>**

**</html>**

**To use <c:forEach> tag as forEach loop we will use the following tag.**

**<c:forEach var=”--” items=”--”>**

**—--**

**</c:forEach>**

**Where “var” attribute is able to take a variable to hold the value at each and every iteration.**

**Where “items attribute is able to take an array reference or a collection reference to get elements.**

**EX:**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<%**

**// Instead of using scriptlet for initializing and setting attributes,**

**// it's better to use JSTL for data manipulation.**

**String[] str = { "AAA", "BBB", "CCC", "DDD", "EEE" };**

**request.setAttribute("strArray", str);**

**%>**

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>JSP forEach Loop with Array</title>**

**</head>**

**<body>**

**<%-- Use JSTL forEach loop to iterate over the array --%>**

**<c:forEach var=*"element"* items=*"*${strArray}*"*>**

**<c:out value=*"*${element}*"* />**

**<br>**

**</c:forEach>**

**</body>**

**</html>**

**<c:forTokens>:**

**------------------------**

**It can be used to perform String tokenization.**

**<c:forTokens var=”--” items=”--” delims=”--”>**

**—-----**

**</c:forTokens>**

**Where “var” attribute will take a variable to hold a token at each and every iteration.**

**Where “items” attribute will take a String to tokenize.**

**Where “delims” attribute is able to take delimiter to perform String tokenization.**

**Ex:**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

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

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>JSP forTokens Loop</title>**

**</head>**

**<body>**

**<%-- Use JSTL forTokens loop to iterate over tokens --%>**

**<c:forTokens var=*"token"* items=*"i am big fan of katrina and sunnyleone"***

**delims=*" "*>**

**<c:out value=*"*${token}*"* />**

**<br>**

**</c:forTokens>**

**</body>**

**</html>**

**<c:import>**

**----------------**

**It can be used to import the content of the target resource into the present JSP page.**

**<c:import url=”--”/>**

**Where “url” attribute is able to take the target resource name and location.**

**EX:**

**<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"***

**pageEncoding=*"UTF-8"*%>**

**<!DOCTYPE html>**

**<html lang=*"en"*>**

**<head>**

**<meta charset=*"UTF-8"*>**

**<title>abc.jsp</title>**

**</head>**

**<body>This is from abc.jsp**

**</body>**

**</html>**

**<c:url>**

**------------------**

**It can be used to represent the specified url.**

**<c:url value=”---”/>**

**Where the “url” attribute is able to take the url value.**

**EX:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<c:url var=*"url"* value=*"http://www.codegnan.com"* />**

**<a href=*'*<c:out value="${url}"/>*'*>codegnan</a>**

**<c:redirect>**

**--------------------**

**It is the same as the send redirect mechanism in Servlets.**

**<c:redirect url=”--”/>**

**Where the “url” attribute is able to take a web application url.**

**EX:**

**------------**

**%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

**<c:redirect url=*"http://www.facebook.com/mallikarjunakollipati"*/>**

**SQL Tags:**

**—---------**

**The main purpose of SQL tags is to connect with the database from JSP pages and to perform database operations from Jsp pages.**

**<sql:setDataSource>**

**<sql:update>**

**<sql:query>**

**<sql:transaction>**

**<sql:param>**

**—----**

**<c:setDataSource>**

**--------------------------**

**It can be used to take all jdbc parameters in order to create connection from jsp page to the Database.**

**<sql:setDataSource driver=”--” url=”--” user=”--” password=”--”/>**

**Where driver, url, user and password attributes are able to take all the jdbc parameters like Driver class name, Driver Url, database User name and database Password.**

**<sql:update>**

**------------------------**

**It can be used to execute the non select sql queries like create, insert, update,delete…....queries.**

**<c:update var=”--”>**

**—---Sql query—---**

**</c:update>**

**Where we are able to provide sql queries in the following two styles.**

**1.** **Statement style**

**2.** **PreparedStatement stype**

**If we have place holders in sql query and to set values to the place holders in sql queries we will use <sql:param> tag**

**<sql:param value=”--”/>**

**Or**

**<sql:param>value</sql:param>**

**Ex: table creation in mysql database in jsp**

**------------------------------------------------------------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

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

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Create Employee Table</title>**

**</head>**

**<body>**

**<h2>Create Employee Table</h2>**

**<!-- Form with button to trigger table creation -->**

**<form method=*"post"*>**

**<button type=*"submit"* name=*"createTable"* value=*"true"*>Create**

**Table</button>**

**</form>**

**<c:if test=*"*${param.createTable == 'true'}*"*>**

**<!-- Set up data source -->**

**<sql:setDataSource var=*"dbSource"* driver=*"com.mysql.cj.jdbc.Driver"***

**url=*"jdbc:mysql://localhost:3306/hibernate"* user=*"root"***

**password=*"root"* />**

**<!-- Create table operation -->**

**<c:catch var=*"error"*>**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount"*>**

**CREATE TABLE emp1 (**

**ENO INT(5) PRIMARY KEY,**

**ENAME CHAR(10),**

**ESAL FLOAT,**

**EADDR CHAR(10)**

**)**

**</sql:update>**

**</c:catch>**

**<!-- Display success or error message -->**

**<c:choose>**

**<c:when test=*"*${not empty error}*"*>**

**<h1 style="color: *red*;">**

**Table Creation Failure:**

**<c:out value=*"*${error}*"* />**

**</h1>**

**</c:when>**

**<c:when test=*"*${rowCount == 0}*"*>**

**<h1>Table Created Successfully</h1>**

**</c:when>**

**<c:otherwise>**

**<h1>Unknown Error Occurred</h1>**

**</c:otherwise>**

**</c:choose>**

**</c:if>**

**</body>**

**</html>**

**Ex: Insert data into mysql database in jsp**

**-----------------------------------------------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

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

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Insert Employee Records</title>**

**</head>**

**<body>**

**<h2>Insert Employee Records</h2>**

**<!-- Form with button to trigger record insertion -->**

**<form method=*"post"*>**

**<button type=*"submit"* name=*"insertRecords"* value=*"true"*>Insert**

**Records</button>**

**</form>**

**<c:if test=*"*${param.insertRecords == 'true'}*"*>**

**<!-- Set up data source -->**

**<sql:setDataSource var=*"dbSource"* driver=*"com.mysql.cj.jdbc.Driver"***

**url=*"jdbc:mysql://localhost:3306/hibernate"* user=*"root"***

**password=*"root"* />**

**<!-- Insert record operations -->**

**<c:catch var=*"error"*>**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount1"***

**sql=*"insert into emp1 values(111,'AAA', 5000, 'Hyd')"* />**

**RowCount1: <c:out value=*"*${rowCount1}*"* />**

**<br>**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount2"*>**

**insert into emp1 values(222, 'BBB', 6000, 'Hyd')**

**</sql:update>**

**RowCount2: <c:out value=*"*${rowCount2}*"* />**

**<br>**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount3"***

**sql=*"insert into emp1 values(?,?,?,?)"*>**

**<sql:param value=*"333"* />**

**<sql:param value=*"CCC"* />**

**<sql:param value=*"7000"* />**

**<sql:param value=*"Hyd"* />**

**</sql:update>**

**RowCount3: <c:out value=*"*${rowCount3}*"* />**

**<br>**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount4"*>**

**insert into emp1 values(?,?,?,?)**

**<sql:param value=*"444"* />**

**<sql:param value=*"DDD"* />**

**<sql:param value=*"8000"* />**

**<sql:param value=*"Hyd"* />**

**</sql:update>**

**RowCount4: <c:out value=*"*${rowCount4}*"* />**

**<br>**

**</c:catch>**

**<!-- Display error message if any -->**

**<c:if test=*"*${not empty error}*"*>**

**<p style="color: *red*;">**

**Error:**

**<c:out value=*"*${error}*"* />**

**</p>**

**</c:if>**

**</c:if>**

**</body>**

**</html>**

**EX: update mysql database employee records by using jsp**

**---------------------------------------------------------------------------------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

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

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Update Employee Salaries</title>**

**</head>**

**<body>**

**<h2>Update Employee Salaries</h2>**

**<!-- Set up data source -->**

**<sql:setDataSource var=*"dbSource"* driver=*"com.mysql.cj.jdbc.Driver"***

**url=*"jdbc:mysql://localhost:3306/hibernate"* user=*"root"***

**password=*"root"* />**

**<!-- Error handling -->**

**<c:catch var=*"error"*>**

**<!-- Update query to increment salary -->**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount"*>**

**UPDATE emp1 SET ESAL = ESAL + 500 WHERE ESAL < 10000**

**</sql:update>**

**</c:catch>**

**<!-- Display results or error message -->**

**<c:choose>**

**<c:when test=*"*${not empty error}*"*>**

**<p style="color: *red*;">**

**Error:**

**<c:out value=*"*${error}*"* />**

**</p>**

**</c:when>**

**<c:otherwise>**

**<p>**

**Number of employees whose salary was updated:**

**<c:out value=*"*${rowCount}*"* />**

**</p>**

**</c:otherwise>**

**</c:choose>**

**</body>**

**</html>**

**EX: delete employee records on mysql by using jsp**

**----------------------------------------------------------------------------**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"***

**pageEncoding=*"ISO-8859-1"*%>**

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

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

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Delete Employees</title>**

**</head>**

**<body>**

**<sql:setDataSource var=*"dbSource"* driver=*"com.mysql.cj.jdbc.Driver"***

**url=*"jdbc:mysql://localhost:3306/adjava"* user=*"root"* password=*"root"* />**

**<c:choose>**

**<c:when test=*"*${not empty param.confirmDelete}*"*>**

**<sql:update dataSource=*"*${dbSource}*"* var=*"rowCount"*>**

**DELETE FROM emp1 WHERE ESAL < 10000**

**</sql:update>**

**<p>**

**Employees Deleted:**

**<c:out value=*"*${rowCount}*"* />**

**</p>**

**</c:when>**

**<c:otherwise>**

**<form method=*"post"* action=*""*>**

**<input type=*"hidden"* name=*"confirmDelete"* value=*"true"* />**

**<p>Are you sure you want to delete employees with ESAL < 10000?</p>**

**<button type=*"submit"*>Yes, delete</button>**

**</form>**

**</c:otherwise>**

**</c:choose>**

**</body>**

**</html>**

**<sql:query>**

**It can be used to retrieve data from the Database table.**

**Syntax-1:**

**<sql:query var=”--” sql=”---”/>**

**Syntax-2:**

**<sql:query var=”--”>**

**—--Sql Query—---**

**</sql:query>**

**If we execute a select sql query with the above tags then the select sql query will be executed and the generated results are available in the form of “result” variable.**

**In the result object, the complete table data is available which includes column names in the form of a single dimensional array and columns data in the form of 2-Dimensional array.**

**Where column names array is represented in the form of “columnNames” predefined variables.**

**Where Column data is represented in the form of “rowsByIndex” predefined variable.**

**EX:**

**<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"* pageEncoding=*"ISO-8859-1"*%>**

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

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

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset=*"ISO-8859-1"*>**

**<title>Employee List</title>**

**</head>**

**<body>**

**<h2>Employee List</h2>**

**<!-- Set up data source -->**

**<sql:setDataSource var=*"dbSource"* driver=*"com.mysql.cj.jdbc.Driver"***

**url=*"jdbc:mysql://localhost:3306/hibernate"***

**user=*"root"* password=*"root"*/>**

**<!-- Query to fetch employee data -->**

**<sql:query dataSource=*"*${dbSource}*"* var=*"result"*>**

**SELECT \* FROM emp1**

**</sql:query>**

**<!-- Error handling -->**

**<c:if test=*"*${result.rowCount == 0}*"*>**

**<p>No employees found.</p>**

**</c:if>**

**<!-- Display results in a table -->**

**<table border=*"1"*>**

**<tr>**

**<c:forEach var=*"columnName"* items=*"*${result.columnNames}*"*>**

**<th><c:out value=*"*${columnName}*"*/></th>**

**</c:forEach>**

**</tr>**

**<c:forEach var=*"row"* items=*"*${result.rowsByIndex}*"*>**

**<tr>**

**<c:forEach var=*"column"* items=*"*${row}*"*>**

**<td><c:out value=*"*${column}*"*/></td>**

**</c:forEach>**

**</tr>**

**</c:forEach>**

**</table>**

**</body>**

**</html>**

**Student Managment Application By using JDBC,Servlets,JSP**

**==================================================================**

**CREATE DATABASE student\_management;**

**USE student\_management;**

**CREATE TABLE `students` (**

**`id` int NOT NULL AUTO\_INCREMENT,**

**`name` varchar(50) NOT NULL,**

**`age` int NOT NULL,**

**`course` varchar(50) NOT NULL,**

**PRIMARY KEY (`id`)**

**) ;**

**CREATE TABLE `users` (**

**`id` int NOT NULL AUTO\_INCREMENT,**

**`username` varchar(50) NOT NULL,**

**`password` varchar(50) NOT NULL,**

**`role` enum('admin','user') NOT NULL,**

**PRIMARY KEY (`id`),**

**UNIQUE KEY `username` (`username`)**

**) ;**

**INSERT INTO `students` (`name`, `age`, `course`) VALUES**

**('Alice Johnson', 20, 'Computer Science'),**

**('Bob Smith', 22, 'Mathematics'),**

**('Charlie Brown', 19, 'Physics'),**

**('Diana Prince', 21, 'Literature');**

**INSERT INTO `users` (`username`, `password`, `role`) VALUES**

**('adminUser', 'securePassword123', 'admin'),**

**('regularUser1', 'password456', 'user'),**

**('johnDoe', 'mypassword789', 'user'),**

**('janeDoe', 'anotherSecurePassword', 'user');**

**Student.java**

**—------------------**

**package com.codegnan.studentapp.model;**

**public class Student {**

**private int id;**

**private String name;**

**private int age;**

**private String course;**

**// Constructors**

**public Student() {**

**}**

**public Student(int id, String name, int age, String course) {**

**this.id = id;**

**this.name = name;**

**this.age = age;**

**this.course = course;**

**}**

**// Getters and Setters**

**public int getId() {**

**return id;**

**}**

**public void setId(int id) {**

**this.id = id;**

**}**

**public String getName() {**

**return name;**

**}**

**public void setName(String name) {**

**this.name = name;**

**}**

**public int getAge() {**

**return age;**

**}**

**public void setAge(int age) {**

**this.age = age;**

**}**

**public String getCourse() {**

**return course;**

**}**

**public void setCourse(String course) {**

**this.course = course;**

**}**

**}**

**login.jsp**

**—--------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Login</title>**

**<style>**

**body {**

**display: *flex*;**

**flex-direction: *column*;**

**justify-content: *center*;**

**align-items: *center*;**

**height: *100vh*;**

**margin: *0*;**

**}**

**table {**

**border-collapse: *collapse*;**

**margin: *auto*;**

**}**

**td {**

**padding: *10px*;**

**text-align: *center*;**

**}**

**h1 {**

**text-align: *center*;**

**}**

**h2 {**

**text-align: *center*;**

**color: *#666*;**

**}**

**form {**

**display: *flex*;**

**flex-direction: *column*;**

**align-items: *center*;**

**}**

**td button {**

**display: *block*;**

**margin: *10px auto 0 auto*;**

**}**

**</style>**

**</head>**

**<body>**

**<h1>Student Management</h1>**

**<h2>Please login to access the system</h2>**

**<form action=*"login"* method=*"post"*>**

**<table>**

**<tr>**

**<td><label for=*"username"*>Username:</label></td>**

**<td><input type=*"text"* id=*"username"* name=*"username"* required></td>**

**</tr>**

**<tr>**

**<td><label for=*"password"*>Password:</label></td>**

**<td><input type=*"password"* id=*"password"* name=*"password"* required></td>**

**</tr>**

**<tr>**

**<td colspan=*"2"*><input type=*"submit"* value=*"Login"*></td>**

**</tr>**

**</table>**

**</form>**

**<%= request.getParameter("error") != null ? request.getParameter("error") : "" %>**

**</body>**

**</html>**

**index.jsp**

**—--------------**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Student Manager</title>**

**<meta http-equiv=*"refresh"* content=*"0;url=login.jsp"*>**

**</head>**

**<body>**

**<!-- Content of your index page goes here -->**

**</body>**

**</html>**

**students.jsp**

**—-----------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* %>**

**<%@ page import=*"java.util.List"* %>**

**<%@ page import=*"com.codegnan.studentapp.model.Student"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Students</title>**

**<style>**

**body {**

**display: *flex*;**

**flex-direction: *column*;**

**justify-content: *center*;**

**align-items: *center*;**

**height: *100vh*;**

**margin: *0*;**

**}**

**table {**

**border-collapse: *collapse*;**

**margin: *auto*;**

**}**

**td {**

**padding: *10px*;**

**}**

**td[colspan="2"] {**

**text-align: *center*;**

**}**

**h1 {**

**text-align: *center*;**

**}**

**form {**

**display: *inline*; /\* Adjusted for inline form display \*/**

**margin-right: *5px*; /\* Added margin for button spacing \*/**

**}**

**button {**

**margin-top: *10px*;**

**}**

**</style>**

**</head>**

**<body>**

**<h1>Student List</h1>**

**<table border=*"1"*>**

**<tr>**

**<th>ID</th>**

**<th>Name</th>**

**<th>Age</th>**

**<th>Course</th>**

**<% if ("admin".equals(session.getAttribute("role"))) { %>**

**<th>Actions</th>**

**<% } %>**

**</tr>**

**<% List<Student> students = (List<Student>) request.getAttribute("students");**

**for (Student student : students) { %>**

**<tr>**

**<td><%= student.getId() %></td>**

**<td><%= student.getName() %></td>**

**<td><%= student.getAge() %></td>**

**<td><%= student.getCourse() %></td>**

**<% if ("admin".equals(session.getAttribute("role"))) { %>**

**<td>**

**<form action=*"edit-student"* method=*"get"*>**

**<input type=*"hidden"* name=*"id"* value=*"*<%= student.getId() %>*"*>**

**<button type=*"submit"*>Edit</button>**

**</form>**

**<form action=*"delete-student"* method=*"post"*>**

**<input type=*"hidden"* name=*"id"* value=*"*<%= student.getId() %>*"*>**

**<button type=*"submit"*>Delete</button>**

**</form>**

**</td>**

**<% } %>**

**</tr>**

**<% } %>**

**</table>**

**<% if ("admin".equals(session.getAttribute("role"))) { %>**

**<a href=*"add-student"*>Add Student</a>**

**<% } %>**

**<a href=*"logout"*>Logout</a>**

**</body>**

**</html>**

**add-student.jsp**

**—-----------------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* %>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Add Student</title>**

**<style>**

**body {**

**display: *flex*;**

**flex-direction: *column*;**

**justify-content: *center*;**

**align-items: *center*;**

**height: *100vh*;**

**margin: *0*;**

**}**

**table {**

**border-collapse: *collapse*;**

**margin: *auto*;**

**}**

**td {**

**padding: *10px*;**

**}**

**td[colspan="2"] {**

**text-align: *center*;**

**}**

**h1 {**

**text-align: *center*;**

**}**

**form {**

**display: *flex*;**

**flex-direction: *column*;**

**align-items: *center*;**

**}**

**button {**

**margin-top: *10px*;**

**}**

**</style>**

**</head>**

**<body>**

**<h1>Add Student</h1>**

**<form action=*"add-student"* method=*"post"*>**

**<table>**

**<tr>**

**<td><label for=*"name"*>Name:</label></td>**

**<td><input type=*"text"* id=*"name"* name=*"name"* required></td>**

**</tr>**

**<tr>**

**<td><label for=*"age"*>Age:</label></td>**

**<td><input type=*"number"* id=*"age"* name=*"age"* required></td>**

**</tr>**

**<tr>**

**<td><label for=*"course"*>Course:</label></td>**

**<td><input type=*"text"* id=*"course"* name=*"course"* required></td>**

**</tr>**

**</table>**

**<button type=*"submit"*>Add</button>**

**</form>**

**<a href=*"students"*>Back to Student List</a>**

**</body>**

**</html>**

**edit-student.jsp**

**—-----------------------------**

**<%@ page contentType=*"text/html;charset=UTF-8"* %>**

**<%@ page import=*"java.util.Map"* %>**

**<%**

**response.setHeader("Cache-Control", "no-cache, no-store, must-revalidate"); // HTTP 1.1**

**response.setHeader("Pragma", "no-cache"); // HTTP 1.0**

**response.setDateHeader("Expires", 0); // Proxies**

**// Session check**

**if (session == null) {**

**response.sendRedirect("login.jsp");**

**return;**

**}**

**String role = (String) session.getAttribute("role");**

**if (role == null) {**

**response.sendRedirect("login.jsp");**

**return;**

**}**

**// Role check**

**if (!"admin".equals(role)) {**

**response.sendRedirect("students");**

**return;**

**}**

**Map<String, String> student = (Map<String, String>) request.getAttribute("student");**

**%>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Edit Student</title>**

**<style>**

**body {**

**display: *flex*;**

**flex-direction: *column*;**

**justify-content: *center*;**

**align-items: *center*;**

**height: *100vh*;**

**margin: *0*;**

**}**

**table {**

**border-collapse: *collapse*;**

**margin: *auto*;**

**}**

**td {**

**padding: *10px*;**

**}**

**td[colspan="2"] {**

**text-align: *center*;**

**}**

**h1 {**

**text-align: *center*;**

**}**

**form {**

**display: *flex*;**

**flex-direction: *column*;**

**align-items: *center*;**

**}**

**button {**

**margin-top: *10px*;**

**}**

**</style>**

**</head>**

**<body>**

**<h1>Edit Student</h1>**

**<%**

**if (student == null) {**

**out.println("<p>Error: No student data available.</p>");**

**} else {**

**%>**

**<form action=*"edit-student"* method=*"post"*>**

**<input type=*"hidden"* name=*"id"* value=*"*<%= student.get("id") %>*"*>**

**<table>**

**<tr>**

**<td><label for=*"name"*>Name:</label></td>**

**<td><input type=*"text"* id=*"name"* name=*"name"* value=*"*<%= student.get("name") %>*"* required></td>**

**</tr>**

**<tr>**

**<td><label for=*"age"*>Age:</label></td>**

**<td><input type=*"number"* id=*"age"* name=*"age"* value=*"*<%= student.get("age") %>*"* required></td>**

**</tr>**

**<tr>**

**<td><label for=*"course"*>Course:</label></td>**

**<td><input type=*"text"* id=*"course"* name=*"course"* value=*"*<%= student.get("course") %>*"* required></td>**

**</tr>**

**<tr>**

**<td colspan=*"2"*><button type=*"submit"*>Update</button></td>**

**</tr>**

**</table>**

**</form>**

**<%**

**}**

**%>**

**<a href=*"students"*>Back to Student List</a>**

**</body>**

**</html>**

**DatabaseUtil**

**—--------------------**

**package com.codegnan.studentapp.util;**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.SQLException;**

**public class DatabaseUtil {**

**private static final String *URL* = "jdbc:mysql://localhost:3306/student\_management";**

**private static final String *USER* = "root";**

**private static final String *PASSWORD* = "root";**

**static {**

**try {**

**// Load the MySQL JDBC driver**

**Class.*forName*("com.mysql.cj.jdbc.Driver");**

**} catch (ClassNotFoundException e) {**

**throw new RuntimeException("Failed to load MySQL driver", e);**

**}**

**}**

**public static Connection getConnection() throws SQLException {**

**return DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*);**

**}**

**}**

**LoginServlet**

**—--------------------**

**package com.codegnan.studentapp.servlets;**

**import java.io.IOException;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import javax.servlet.http.HttpSession;**

**import com.codegnan.studentapp.util.DatabaseUtil;**

**@WebServlet("/login")**

**public class LoginServlet extends HttpServlet {**

**protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**String username = request.getParameter("username");**

**String password = request.getParameter("password");**

**boolean loggedIn = false;**

**try (Connection connection = DatabaseUtil.*getConnection*()) {**

**String sql = "SELECT \* FROM users WHERE username = ? AND password = ?";**

**PreparedStatement statement = connection.prepareStatement(sql);**

**statement.setString(1, username);**

**statement.setString(2, password);**

**ResultSet resultSet = statement.executeQuery();**

**if (resultSet.next()) {**

**String role = resultSet.getString("role");**

**HttpSession session = request.getSession();**

**session.setAttribute("username", username);**

**session.setAttribute("role", role);**

**// Set loggedIn to true since the user has successfully logged in**

**loggedIn = true;**

**session.setAttribute("loggedIn", loggedIn);**

**response.sendRedirect("students");**

**} else {**

**response.sendRedirect("login.jsp?error=Invalid username or password");**

**}**

**} catch (SQLException e) {**

**throw new ServletException(e);**

**}**

**}**

**}**

**LogoutServlet**

**—-------------------------**

**package com.codegnan.studentapp.servlets;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import javax.servlet.http.HttpSession;**

**import java.io.IOException;**

**@WebServlet("/logout")**

**public class LogoutServlet extends HttpServlet {**

**protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**HttpSession session = request.getSession();**

**System.*err*.println("Invalidating session");**

**session.invalidate();**

**response.sendRedirect("login.jsp");**

**}**

**}**

**AddStudentServlet**

**—--------------------------**

**package com.codegnan.studentapp.servlets;**

**import java.io.IOException;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.util.HashMap;**

**import java.util.Map;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import javax.servlet.http.HttpSession;**

**import com.codegnan.studentapp.util.DatabaseUtil;**

**@WebServlet("/add-student")**

**public class AddStudentServlet extends HttpServlet {**

**protected void doGet(HttpServletRequest request, HttpServletResponse response)**

**throws ServletException, IOException {**

**HttpSession session = request.getSession(false);**

**Boolean loggedIn = (Boolean) session.getAttribute("loggedIn");**

**String role = (String) session.getAttribute("role");**

**if (session == null || loggedIn == null || !loggedIn) {**

**response.sendRedirect("login.jsp");**

**return;**

**}**

**else if ("user".equals(role)) {**

**response.sendRedirect("students");**

**return;**

**} else {**

**request.getRequestDispatcher("add-student.jsp").forward(request, response);**

**}**

**}**

**protected void doPost(HttpServletRequest request, HttpServletResponse response)**

**throws ServletException, IOException {**

**HttpSession session = request.getSession();**

**Boolean loggedIn = (Boolean) session.getAttribute("loggedIn");**

**String role = (String) session.getAttribute("role");**

**if (session == null || loggedIn == null || !loggedIn) {**

**response.sendRedirect("login.jsp");**

**return;**

**}**

**if (!"admin".equals(role)) {**

**response.sendRedirect("students");**

**return;**

**}**

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

**String age = request.getParameter("age");**

**String course = request.getParameter("course");**

**try (Connection connection = DatabaseUtil.*getConnection*()) {**

**String sql = "INSERT INTO students (name, age, course) VALUES (?, ?, ?)";**

**PreparedStatement statement = connection.prepareStatement(sql);**

**statement.setString(1, name);**

**statement.setInt(2, Integer.*parseInt*(age));**

**statement.setString(3, course);**

**statement.executeUpdate();**

**response.sendRedirect("students");**

**} catch (SQLException e) {**

**throw new ServletException(e);**

**}**

**}**

**}**

**DeleteStudentServlet**

**—----------------------------------**

**package com.codegnan.studentapp.servlets;**

**import java.io.IOException;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.SQLException;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import javax.servlet.http.HttpSession;**

**import com.codegnan.studentapp.util.DatabaseUtil;**

**@WebServlet("/delete-student")**

**public class DeleteStudentServlet extends HttpServlet {**

**protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**HttpSession session = request.getSession();**

**String role = (String) session.getAttribute("role");**

**if (!"admin".equals(role)) {**

**response.sendRedirect("students");**

**return;**

**}**

**String id = request.getParameter("id");**

**try (Connection connection = DatabaseUtil.*getConnection*()) {**

**String sql = "DELETE FROM students WHERE id = ?";**

**PreparedStatement statement = connection.prepareStatement(sql);**

**statement.setInt(1, Integer.*parseInt*(id));**

**statement.executeUpdate();**

**response.sendRedirect("students");**

**} catch (SQLException e) {**

**throw new ServletException(e);**

**}**

**}**

**}**

**EditStudentServlet**

**—-----------------------------**

**package com.codegnan.studentapp.servlets;**

**import java.io.IOException;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.util.HashMap;**

**import java.util.Map;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import javax.servlet.http.HttpSession;**

**import com.codegnan.studentapp.util.DatabaseUtil;**

**@WebServlet("/edit-student")**

**public class EditStudentServlet extends HttpServlet {**

**protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**HttpSession session = request.getSession(false);**

**Boolean loggedIn = (Boolean) session.getAttribute("loggedIn");**

**String role = (String) session.getAttribute("role");**

**if (session == null || loggedIn == null || !loggedIn) {**

**response.sendRedirect("login.jsp");**

**return;**

**}**

**if ("user".equals(role)) {**

**response.sendRedirect("students");**

**return;**

**}**

**String id = request.getParameter("id");**

**try (Connection connection = DatabaseUtil.*getConnection*()) {**

**String sql = "SELECT \* FROM students WHERE id = ?";**

**PreparedStatement statement = connection.prepareStatement(sql);**

**statement.setInt(1, Integer.*parseInt*(id));**

**ResultSet resultSet = statement.executeQuery();**

**if (resultSet.next()) {**

**Map<String, String> student = new HashMap<>();**

**student.put("id", String.*valueOf*(resultSet.getInt("id")));**

**student.put("name", resultSet.getString("name"));**

**student.put("age", String.*valueOf*(resultSet.getInt("age")));**

**student.put("course", resultSet.getString("course"));**

**request.setAttribute("student", student);**

**request.getRequestDispatcher("edit-student.jsp").forward(request, response);**

**} else {**

**response.sendRedirect("students");**

**}**

**} catch (SQLException e) {**

**throw new ServletException(e);**

**}**

**}**

**protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {**

**HttpSession session = request.getSession(false);**

**Boolean loggedIn = (Boolean) session.getAttribute("loggedIn");**

**String role = (String) session.getAttribute("role");**

**if (session == null || loggedIn == null || !loggedIn || !"admin".equals(role)) {**

**response.sendRedirect("students");**

**return;**

**}else if(!loggedIn) {**

**response.sendRedirect("login");**

**}**

**String id = request.getParameter("id");**

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

**String age = request.getParameter("age");**

**String course = request.getParameter("course");**

**try (Connection connection = DatabaseUtil.*getConnection*()) {**

**String sql = "UPDATE students SET name = ?, age = ?, course = ? WHERE id = ?";**

**PreparedStatement statement = connection.prepareStatement(sql);**

**statement.setString(1, name);**

**statement.setInt(2, Integer.*parseInt*(age));**

**statement.setString(3, course);**

**statement.setInt(4, Integer.*parseInt*(id));**

**statement.executeUpdate();**

**response.sendRedirect("students");**

**} catch (SQLException e) {**

**throw new ServletException(e);**

**}**

**}**

**}**

**ViewStudentsServlet**

**—------------------------------**

**package com.codegnan.studentapp.servlets;**

**import java.io.IOException;**

**import java.sql.Connection;**

**import java.sql.PreparedStatement;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.util.ArrayList;**

**import java.util.List;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import com.codegnan.studentapp.model.Student;**

**import com.codegnan.studentapp.util.DatabaseUtil;**

**@WebServlet("/students")**

**public class ViewStudentsServlet extends HttpServlet {**

**protected void doGet(HttpServletRequest request, HttpServletResponse response)**

**throws ServletException, IOException {**

**List<Student> students = new ArrayList<>();**

**try (Connection connection = DatabaseUtil.*getConnection*()) {**

**String sql = "SELECT \* FROM students";**

**PreparedStatement statement = connection.prepareStatement(sql);**

**ResultSet resultSet = statement.executeQuery();**

**while (resultSet.next()) {**

**int id = resultSet.getInt("id");**

**String name = resultSet.getString("name");**

**int age = resultSet.getInt("age");**

**String course = resultSet.getString("course");**

**Student student = new Student(id, name, age, course);**

**students.add(student);**

**}**

**} catch (SQLException e) {**

**throw new ServletException(e);**

**}**

**request.setAttribute("students", students);**

**request.getRequestDispatcher("students.jsp").forward(request, response);**

**}**

**}**