ICON OF JAVA

ADV. JAVA

Servlets

8.Servlet Communication

Mr. Nagoor Babu M.Tech

[ICON of JAVA]

(Sun certified & Realtime Expert)

Ex. HCL Employee

Trained Lakhs of Students
for last 14 years across INDIA

India's No.1 Software Training Institute

DURGASOFT

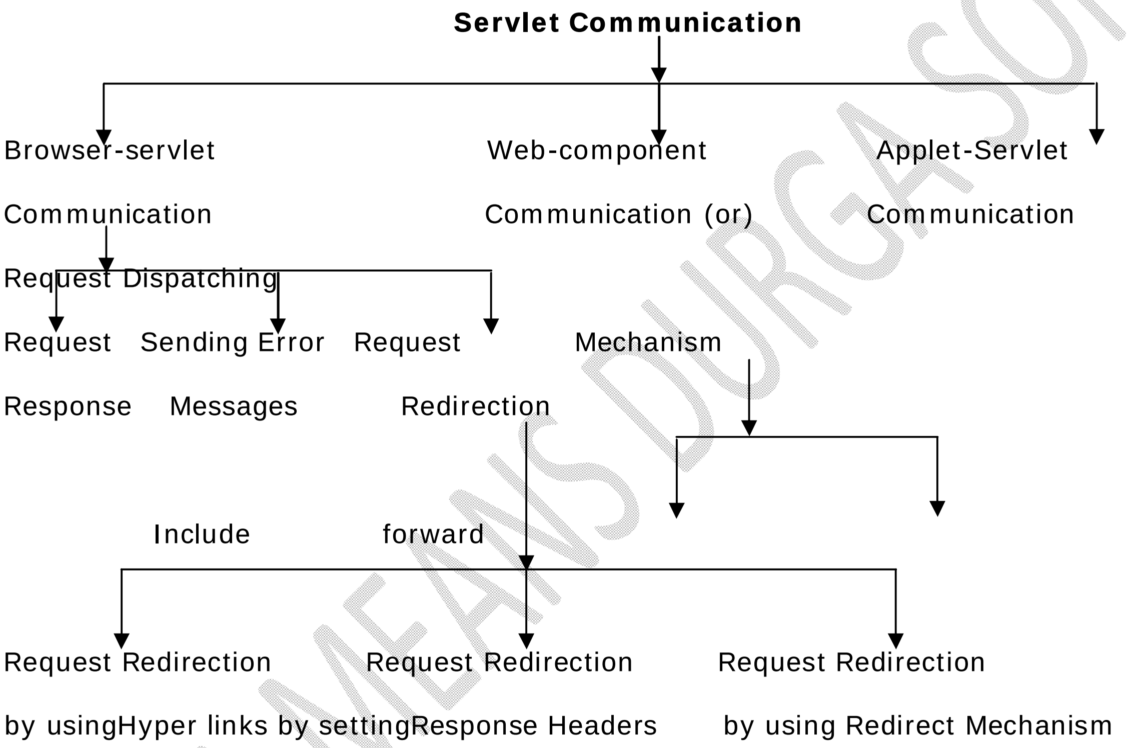
www.durgasoft.com Ph: 9246212143 ,8096969696

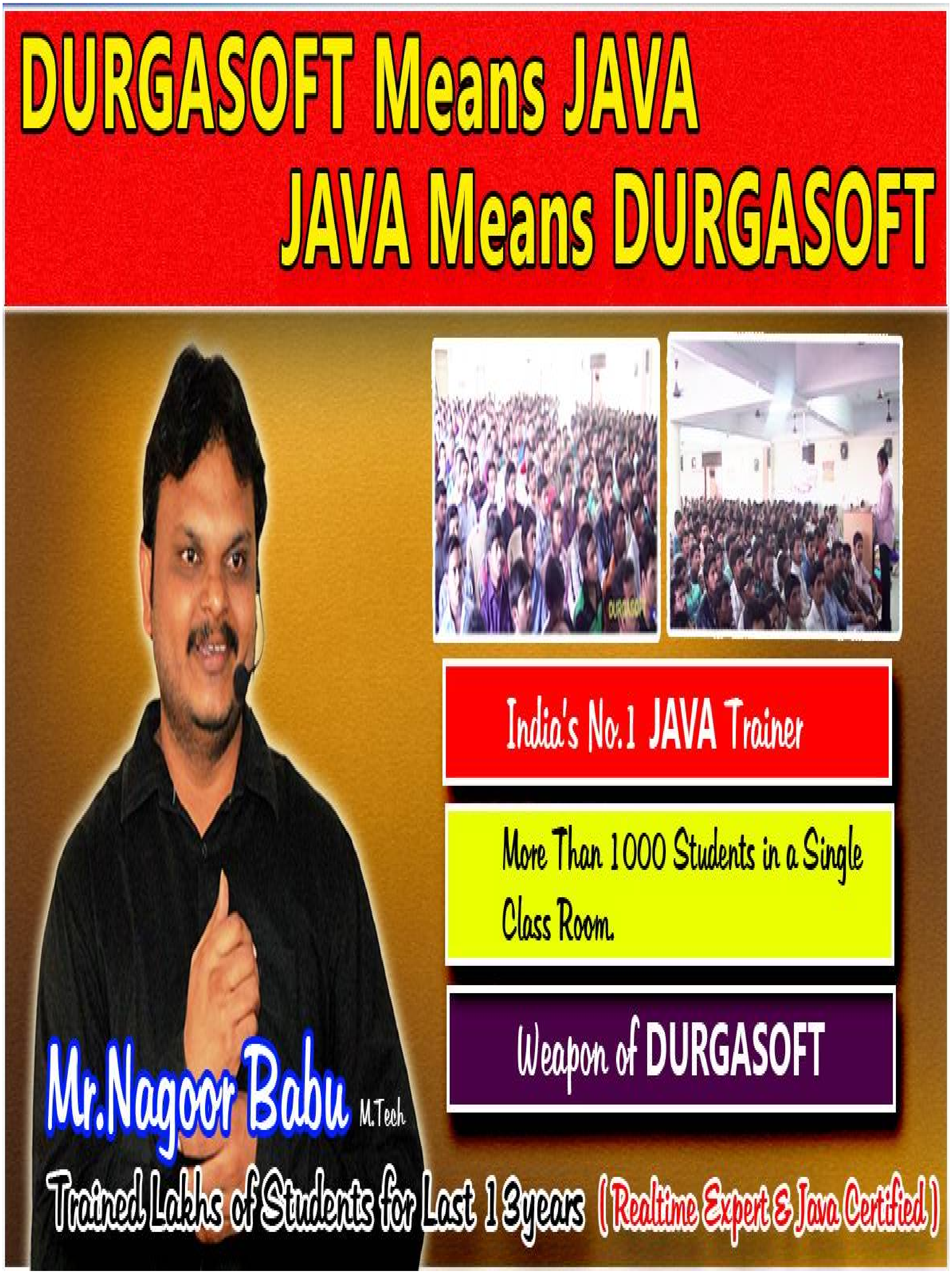
Servlet Communication

In general in web application deployment is not at all suggestible to provide the complete application logic with in a single web resource, it is suggestible to distribute the complete application logic over multiple web resources.

In the above context, to execute the application we must require communication between all the web resources, for this we have to use Servlet Communication.

In web application, we are able to provide servlet communication in the following 3 ways.


Servlet Communication
Browser-servlet Communication
Web-component Communication (or) Request Dispatching Mechanism
Applet-Servlet Communication
Request Response
Sending Error Messages
Request Redirection
Include
forward
Request Redirection by usingHyper links
Request Redirection by settingResponse Headers
Request Redirection by using Redirect Mechanism


DURGASOFT Means JAVA
JAVA Means DURGASOFT
India's No.1 JAVA Trainer
More Than 1000 Students in a Single Class Room.
Weapon of DURGASOFT
Mr.Nagoor Babu M.Tech
Trained Lakhs of Students for Last 13years (Realtime Expert & Java Certified)

1. Browser-Servlet Communication:

In general in web applications, we will use browser as a client at client machine, from the browser we will send a request to a servlet available at server, where the servlet will be executed and generate some response to the client browser.

In the above process, we have provided communication between client browser and servlet, so that sending a normal request from client to server and getting a normal response from server to client is an examplefor **Browser-Servlet Communication**.

2. Sending Error Messages:

As part of the web application execution, if the container identify any exception or error then container will send the respective error message to be client in its standalone template.

As part of our application, if we want to send our own messages to the client in the container defined template we have to use the following method from response.

```
        public void sendError(int statuscode, String description)
```

where statuscode may be 5xx.

SendErrorApp:

Registrationform.html

```
<!DOCTYPE html>

<html>

<head>

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

<title>Insert title here</title>

</head>

<body>

<font color='red'>
```

```
<h2>Durga Consultency Services</h2>

<h3>User Registration Form</h3>

</font>

<form method="POST" action="./reg">

<table>

<tr>

        <td>User Name</td>

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

</tr>

<tr>

        <td>User Age</td>

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

</tr>

<tr>

        <td>User Email</td>

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

</tr>

<tr>

        <td>User Mobile</td>

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

</tr>

<tr>

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

</tr>



</table>
```

```
</form>

</body>

</html>
```

Web.xml

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

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">

<display-name>senderrorapp</display-name>

<welcome-file-list>

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

</welcome-file-list>

<servlet>

<description></description>

<display-name>RegistrationServlet</display-name>

<servlet-name>RegistrationServlet</servlet-name>

<servlet-class>com.durgasoft.RegistrationServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>RegistrationServlet</servlet-name>

<url-pattern>/reg</url-pattern>

</servlet-mapping>

</web-app>
```

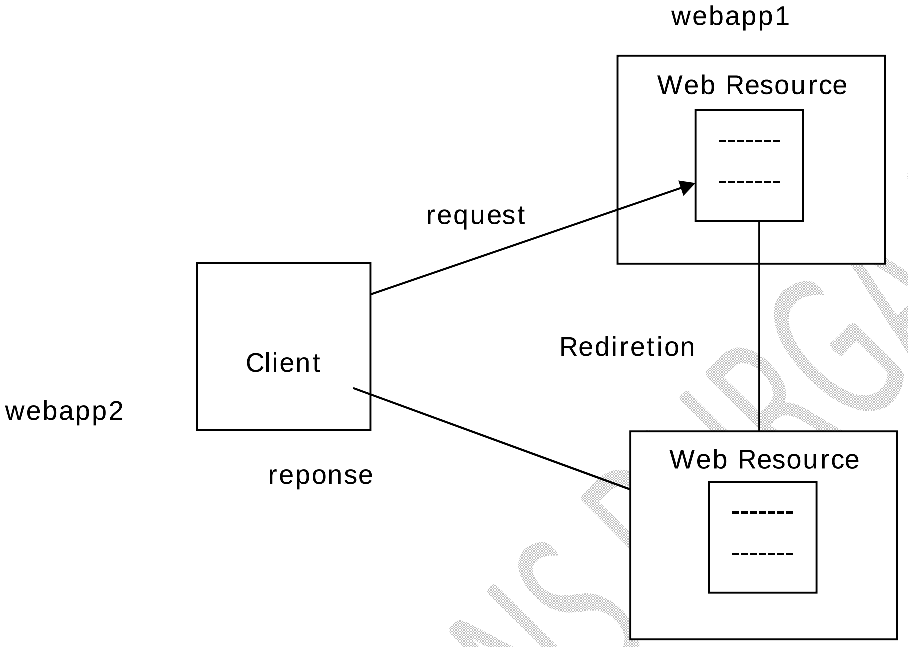
RegistrationServlet.java

```
package com.durgasoft;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class RegistrationServlet extends HttpServlet {
        private static final long serialVersionUID = 1L;
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                try {
                        response.setContentType("text/html");
                        PrintWriter out=response.getWriter();
                        String uname=request.getParameter("uname");
                        int uage=Integer.parseInt(request.getParameter("uage"));
                        String uemail=request.getParameter("uemail");
                        String umobile=request.getParameter("umobile");

                        if(uage<18 || uage>30){
                        response.sendError(504, "User Age Is Not Sufficient for this
                        Recruitment");
                        }else{
                                out.println("<html>");
                                out.println("<body>");
                                out.println("<font color='red'>");
```

```
                    out.println("<h2>Durga Consultency Services</h2>");

                    out.println("<h3>User Registration Details</h3>");

                    out.println("</font>");

                    out.println("<table border='1'>");

                    out.println("<tr><td>User Name</td><td>"+uname+"</td></tr>");

                    out.println("<tr><td>User Age</td><td>"+uage+"</td></tr>");

                    out.println("<tr><td>User Email</td><td>"+uemail+"</td></tr>");

                    out.println("<tr><td>User Mobile</td><td>"+umobile+"</td></tr>");

                    out.println("</table></body></html>");

                }

            } catch (Exception e) {

                e.printStackTrace();

            }

        }


    }
```

3. Request Redirection:

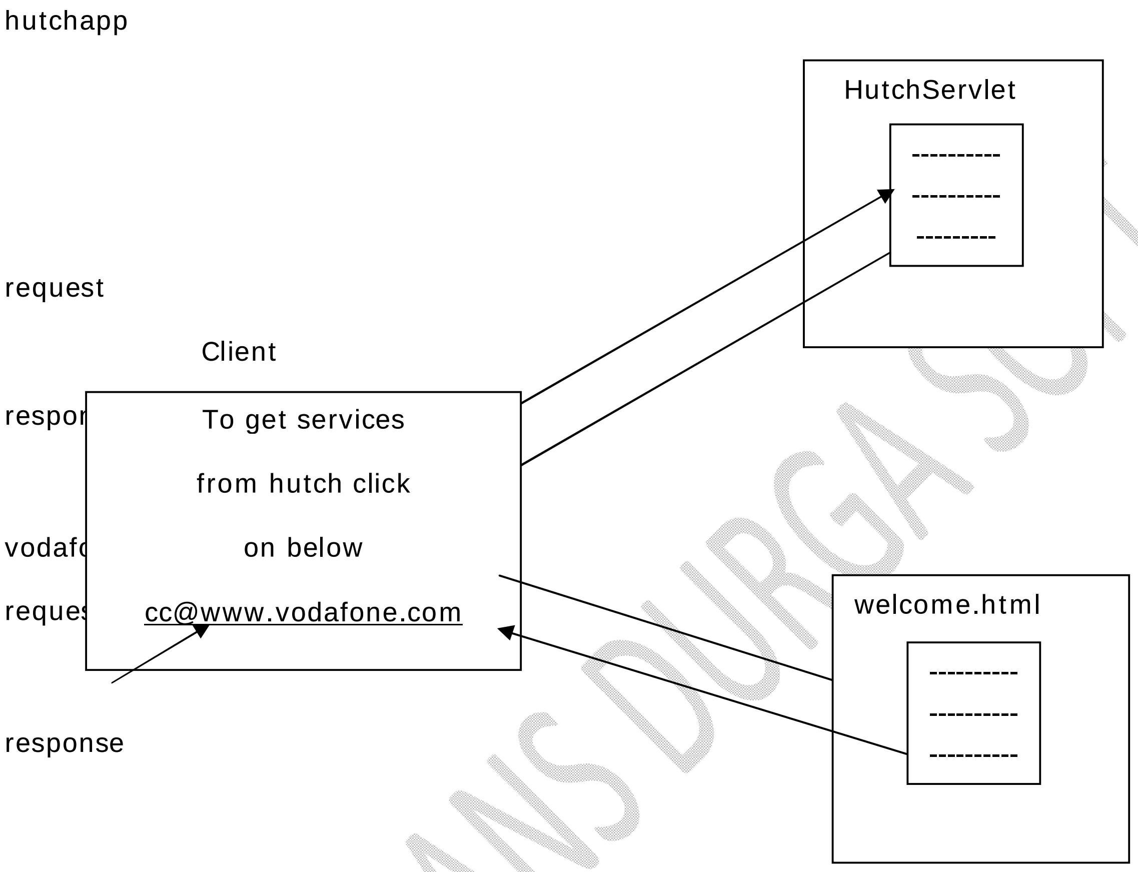
The process of bypassing the request from one web application to another web application is called as **Request Redirection**.



In web applications, we are able to achieve Request Redirection in the following 3 ways.

1. Request Redirection by using Hyper links
2. Request Redirection by setting Response Headers
3. Request Redirection by using Send Redirect Mechanism

1. Request Redirection by using Hyper links:



In this mechanism, when we send a request to first application some resources will be executed and generated an hyper link at client browser as a response.

By getting hyper link at client browser we have to click on it and we have to send another request to the new web application.

By executing some resources at new web application the required response will be generated to client browser.

Drawback:

In this Request Redirection mechanism, user may or may not click the generated hyper link at client browser after sending first request. So that this mechanism won't provide guarantee to achieve Request Redirection.

2. Request Redirection by setting Response Headers:

In this mechanism, first we will send a request to first web application, where first web application will set Redirectional Status Code to Status Line field and new web application URl to Location Response Header.

When the Response Format reached to the client then client will pick up Redirectional status code value from Status Line field, with this client browser will pick up Location Response Header value i.e. new web application URL then client browser will send a new request to new web application.

By executing some resources at new web application the required response will be generated at client machine.

To represent Request Redirection HttpServletResponse has introduced the following 2 constants.

1. public static final int SC_MOVED_TEMPORARILY;
2. public static final int SC_MOVED_PERMANENTLY;

To set a particular status code value to Response Header we will use the following method.

public void setStatus(int statuscode)

To set a particular Response Header value in Response Format we have to use the following method.

public void setHeader(String header_name, String value)

Drawback:

To perform Request Redirection, if we use this approach then every time we have to set Redirectional status code and new web application URL to Location Response Header.

3. Request Redirection by using Send Redirect Mechanism:

To perform Request Redirection, If we use Send Redirect Mechanism no need to use Hyper links, not required to set status code and Response Header values to the Response Format, but We need to use the following method.

public void sendRedirect(String url)

sendredirectapp:-

web.xml:

```
<web-app>
<display-name>sendredirectapp</display-name>
<servlet>
<servlet-name>HutchServlet</servlet-name>
<servlet-class>HutchServlet</servlet-class>
</servlet>


<servlet-mapping>
<servlet-name>HutchServlet</servlet-name>
<url-pattern>/hutch</url-pattern>
</servlet-mapping>
</web-app>
```

HutchServlet.java:

```
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

publicclass HutchServlet extends HttpServlet {

        protectedvoid doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		response.sendRedirect("http://localhost:2020/vodaphone/welcome.html");
	}

}
```

vodaphoneapp:-

welcome.html:

```
<html>
<bodybgcolor="red">
	<center><b><fontsize="7"color="black">
	<br><br>
	Welcome to Vodaphone
	</font></b></center>
</body>
</html>
```

2. Web-Component Communication:

The process of providing communication between more than one web component available at server machine is called as **Web-Component Communication.**

In general, web-component communication is available in between Servlet-Servlet, Servlet-Jsp, Servlet-HTML, Jsp-Jsp, Jsp-Servlet, Jsp-HTML and so on.

In web applications, we are able to achieve web-component communication in the following 2 ways.

1. Include Mechanism
2. Forward Mechanism

If we want to perform the above mechanisms internally we must use RequestDispatcher object. So that both include and forward mechanisms are commonly called as Request Dispatching Mechanisms.

To achieve web-component communication in web applications we have to use the following 2 steps

Step 1: Get RequestDispatcher object.

Step 2: Apply either include or forward mechanism by using the respective methods.

Step 1: RequestDispatcher object:

RequestDispatcher is an object, it will provide very good environment either to include the target resource response into the present resource response or to forward request from present resource to the target resource.

To get RequestDispatcher object we will use the following 2 ways.

1. ServletContext
    1. getRequestDispatcher(_) method
    2. gteNamedDispatcher(_) method
2. ServletRequest
    1. getRequestDispatcher(_) method

Q: What is the difference between getRequestDispatcher(_) method and getNamedDispatcher(_) method from ServletContext ?

Ans: Both the methods are used to get the RequestDispatcher object.

To get RequestDispatcher object, if we use getRequestDispatcher(_) method then we should pass the locator of target resource as parameter.

Note : In case of the servlet, url pattern is treated as locator.

public RequestDispatcher getRequestDispatcher(String path)

To get RequestDispatcher object, if we use getNamedDispatcher(_) method then we have to pass logical name of target resource as parameter.

Note : In caseof the servlet, logical name is a name specified along with <servlet-name> tag in web.xml file.

public RequestDispatcher getNamedDispatcher(String path)

Q: What is the difference between getRequestDispatcher(_) method ServletContext and ServletRequest?

Ans: Both the methods can be used to get the RequestDispatcher object.

To get RequestDispatcher object, if we use getRequestDispatcher(_) method from ServletContext then we must pass the relative path of target resource.

To get RequestDispatcher object, if we use getRequestDispatcher(_) method from ServletRequest then we have to pass either relative path or absolute path of target resource.

Note: In general, relative path should prefix with forward slash("/") and absolute path should not prefix with forward slash("/").

Step 2: Apply either Include mechanism or Forward mechanism to achieve Web-Component Communication:

To represent Include and Forward mechanisms RequestDispatcher has provided the following methods.

public void include(ServletRequest req, ServletResponse res)throws SE, IOE

public void forward(ServletRequest req, ServletResponse res)throws SE, IOE

Q: What are the differences between Include and Forward mechanisms?

Ans: In web applications, Include Request Dispatching mechanism can be used to include the target resource response into the present resource response.

In case of Include mechanism, when we send a request to first resource then container will prepare request and response objects, by executing some part in first resource container may generate some response in response object.

When container encounter include(_,_) method then container will bypass the request and response objects to target resource along with flow of execution without refreshing response object.

By executing the target resource some response will be generated in response object, at the end of target resource container will bypass request and response objects back to the first resource to continue its further execution.

In the above context, container will execute remaining content in first resource , some response will be added to response object, at the end of first resource container will dispatch ovaerall response to client.

Therefore, In case of Include mechanism, client is able to receive all resources which are participated in the present request processing.

In web applications, the main purpose of Forward mechanism is to forward request from present resource to target resource.

In case of Forward mechanism, when we send a request to first resource then container will create request and response objects, by executing some part in first resource container may generate some response in response object.

When container encounter forward(_,_) method then container will bypass the request and response objects to the target resource along with flow of execution by refreshing response object (by eliminating previous content in response object).

By executing the target resource some response will be generated in response object, at the end of target resource container will dispatch ovaerall response to client directly without moving back to the first resource.

Therefore, In case of Forward mechanism, client is able to receive only the target resource response which has included in the present request processing.

Q: What is Servlet Chaining?

Ans: The processofincluding more than one servlet in order to process a single request is called as **Servlet Chaining**or **Servlet Collaboration**.

Q:What are the differences between Forward mechanism and send Redirect mechanism?

Ans: 1. In web applications, Forward Request Dispatching mechanism can be used to provide the communication between two resources which must be available at same server.

In web applications, Send Redirect mechanism can be used to provide the communication between two resources which may be available at same server or at two different servers.

2. In case of Forward mechanism, one request is sufficient to establish the communication between two resources.

In case of Send Redirect mechanism, we need requests to establish the communication between two web resources.

includeapp:-

addform.html:

```
<html>
<bodybgcolor="lightgreen">
<center>
<formmethod="get"action="./add">
<br><br>
<tablebgcolor="lightyellow">
<tr><tdcolspan="2"><center><b><fontsize="6"color="red"><u>Product Add
Form</u></font></b></center></td></tr>
<tr><td>Product Id</td><td><inputtype="text"name="pid"/></td>
</tr>


<tr>
<td>Product Name</td>
<td><inputtype="text"name="pname"/></td>
</tr>
<tr>
<td>Product Cost</td>
<td><inputtype="text"name="pcost"/></td>
</tr>
<tr>
<td><inputtype="submit"value="ADD"/></td>
</tr>
</table></form></center></body></html>
```

web.xml:

```
<web-app>
<display-name>includeapp</display-name>
<welcome-file-list>
<welcome-file>addform.html</welcome-file>
</welcome-file-list>
<servlet>
<description></description>
<display-name>AddServlet</display-name>
<servlet-name>AddServlet</servlet-name>
<servlet-class>AddServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>AddServlet</servlet-name>
<url-pattern>/add</url-pattern>
</servlet-mapping>
</web-app>
```

Product.java:

```
publicclassProduct {
	private String pid;
	private String pname;
	privateintpcost;
	public String getPid() {
		returnpid;
	}
	publicvoid setPid(String pid) {
		this.pid = pid;
	}
	public String getPname() {
		returnpname;
	}
	publicvoid setPname(String pname) {
		this.pname = pname;
	}



publicint getPcost() {
		returnpcost;
	}
	publicvoid setPcost(int pcost) {
		this.pcost = pcost;
	}
}
```

ProductDao.java:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;

publicclass ProductDao {
        Connection con;
        Statement st;
        ResultSet rs;
        ArrayList<Product>al;
        ProductDao(){
                try {
                        Class.forName("oracle.jdbc.driver.OracleDriver");
        con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system",
"durga");
                        st=con.createStatement();
                        al=new ArrayList<Product>();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
        publicvoid add(String pid, String pname, int pcost){
                try {
                        st.executeUpdate("insert into product
values('"+pid+"','"+pname+"',"+pcost+")");
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
        public ArrayList<Product> getProducts(){
                try {
                        rs=st.executeQuery("select * from product");
                        while(rs.next()){
                                Product p=new Product();
                                p.setPid(rs.getString(1));
                                p.setPname(rs.getString(2));
                                p.setPcost(rs.getInt(3));
                                al.add(p);
                        }
                } catch (Exception e) {


                e.printStackTrace();
                }
                returnal;
        }
}
```

AddServlet.java:

```
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

publicclass AddServlet extends HttpServlet {

        protectedvoid doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                try {
                        response.setContentType("text/html");
                        PrintWriter out=response.getWriter();
                        String pid=request.getParameter("pid");
                        String pname=request.getParameter("pname");
                        int pcost=Integer.parseInt(request.getParameter("pcost"));
                        ProductDao pd=new ProductDao();
                        pd.add(pid,pname,pcost);
                        ArrayList<Product> prds=pd.getProducts();
                        out.println("<html><body><center><br><br>");
                        out.println("<table border='1' bgcolor='lightyellow'>");
                out.println("<tr><td>PID</td><td>PNAME</td><td>PCOST</td></tr>");
                        for(Object o : prds){
                                Product p=(Product)o;
        out.println("<tr><td>"+p.getPid()+"</td><td>"+p.getPname()+"</td><td>"+p.g
etPcost()+"</td></tr>");
                        }
                        out.println("</table></center><br><hr><br></body></html>");

        RequestDispatcherrd=request.getRequestDispatcher("/addform.html");
                                rd.include(request, response);
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

forwardapp:-

layout.html:

```
<html>
<framesetrows="20%,65%,15%">
<framesrc="header.html"/>
<framesetcols="20%,80%">
<framesrc="menu.html"/>
<framesrc="welcome.html"name="body"/>
</frameset>
<framesrc="footer.html"/>
</frameset>
</html>
```

Header.html:

```
<html><bodybgcolor="red">
<center><b><fontsize="7"color="white">
    Durga Software Solutions
</font></b></center>
</body></html>
```

menu.html:

```
<html>
<bodybgcolor="cyan"><br><br><center><b><fontsize="6">
<ahref="./addform.html"target="body">Add</a><br><br>
<ahref="./searchform.html"target="body">Search</a> <br><br>
<ahref="./updateform.html"target="body">Update</a><br><br>
<ahref="./deleteform.html"target="body">Delete</a>
</font></b></center></body>
</html>
```

footer.html:

```
<html><bodybgcolor="blue">
<center><b><fontsize="6"color="white">
copyrights2010-2020@www.durgasoft.com
</font></b></center></body></html>
```

welcome.html:

```
<html><bodybgcolor="lightyellow">
<center><b><fontsize="7"color="red">
<br><br><marquee>
Welcome to Durga Software Solutions
</marquee>
</font></b></center></body></html>
```

addform.html:

```
<html>
<bodybgcolor="lightgreen">
<b><fontsize="7">
<br>
<formmethod="get"action="./add">
<pre>
                Student Id          <inputtype="text"name="sid"/>

                Student Name        <inputtype="text"name="sname"/>

                Student Marks       <inputtype="text"name="smarks"/>

                <inputtype="submit"value="Add"/>
</pre></form></font></b></body></html>
```

searchform.html:

```
<html>
<bodybgcolor="lightgreen">
<b><fontsize="7"><br><br>
<formmethod="get"action="./search">
<pre>
                Student Id          <inputtype="text"name="sid"/>

                <inputtype="submit"value="Search"/>
</pre></form></font></b></body></html>
```

updateform.html:

```
<html>
<bodybgcolor="lightgreen">
<b><fontsize="7"><br><br>
<formmethod="get"action="./edit">
<pre>
                Student Id          <inputtype="text"name="sid"/>

                <inputtype="submit"value="GetEditForm"/>
</pre></form></font></b></body></html>
```

deleteform.html:

```
<html>
<bodybgcolor="lightgreen">
<b><fontsize="7">
<br><br>
<formmethod="get"action="./delete">
```

```
<pre>

                Student Id            <inputtype="text"name="sid"/>


<inputtype="submit"value="Delete"/>
</pre></form></font></b></body></html>
```

success.html:

```
<html>
<bodybgcolor="lightyellow">
<center><b><fontsize="7"color="red">
<br><br>
Success
</font></b></center></body></html>
```

failure.html:

```
<html>
<bodybgcolor="lightyellow">
<center><b><fontsize="7"color="red">
<br><br>
Failure
</font></b></center></body></html>
```

existed.html:

```
<html>
<bodybgcolor="lightyellow">
<center><b><fontsize="7"color="red">
<br><br>
Student Existed Already
</font></b></center></body></html>
```

notexisted.html:

```
<html>
<bodybgcolor="lightyellow">
<center><b><fontsize="7"color="red">
<br><br>
Student Not Existed
</font></b></center></body></html>
```

StudentDao.java:

```
import java.sql.ResultSet;

publicinterface StudentDao {
        public String add(String sid, String sname, int smarks);
        public ResultSet search(String sid);
```

```
    public ResultSet getStudent(String sid);
    public String update(String sid, String sname, int smarks);
    public String delete(String sid);
}
```

StudentDaoImpl:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class StudentDaoImpl implements StudentDao {

    Connection con;
    Statement st;
    ResultSet rs;
    String status = "";

    public StudentDaoImpl() {
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            con = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe", "system", "durga");
            st = con.createStatement();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public ResultSet getStudent(String sid) {
        try {
            rs = st.executeQuery("select * from student where sid='" + sid
                    + "'");
        } catch (Exception e) {
            e.printStackTrace();
        }
        return rs;
    }

    public String add(String sid, String sname, int smarks) {
        try {
            rs = getStudent(sid);
            boolean b = rs.next();
            if (b == true) {
                status = "existed";
            } else {
                int rowCount = st.executeUpdate("insert into student values('"
                        + sid + "','" + sname + "'," + smarks + ")");
                if (rowCount == 1) {
                    status = "success";
                } else {
                    status = "failure";
                }
            }
        } catch (Exception e) {


e.printStackTrace();
        }
        return status;
    }

    public ResultSet search(String sid) {
        return getStudent(sid);
    }

    public String update(String sid, String sname, int smarks) {
        try {
            int rowCount = st
                    .executeUpdate("update student set sname='" + sname
                            + "',smarks=" + smarks + " where sid='"
+ sid + "'");
            if (rowCount == 1) {
                status = "success";
            } else {
                status = "failure";
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return status;
    }

    public String delete(String sid) {
        try {
            rs = getStudent(sid);
            boolean b = rs.next();
            if (b == true) {
                int rowCount = st
                .executeUpdate("delete from student where sid='" + sid+ "'");
                if (rowCount == 1) {
                    status = "success";
                } else {
                    status = "failure";
                }
            } else {
                status = "notexisted";
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
                return status;
        }
}
```

AddServlet.java:

```
import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class AddServlet extends HttpServlet
{
        public void doGet(HttpServletRequest req,HttpServletResponse res)throws
ServletException,IOException
        {
                try
                {
                        String sid=req.getParameter("sid");
                        String sname=req.getParameter("sname");
                        int smarks=Integer.parseInt(req.getParameter("smarks"));
                        StudentDao sd=new StudentDaoImpl();
                        String status=sd.add(sid,sname,smarks);
                        if(status.equals("existed"))
                        {
                        RequestDispatcherrd1=req.getRequestDispatcher("/existed.html");
                                rd1.forward(req,res);
                        }
                        if(status.equals("success"))
                        {
                                req.getRequestDispatcher("success.html").forward(req,res);
                        }
                        if(status.equals("failure"))
                        {
                                req.getRequestDispatcher("failure.html").forward(req,res);
                        }
                }
                catch(Exception e)
                {
                        e.printStackTrace();
                }
        }
}
```

SearchServlet.java:

```
import java.io.IOException;
import java.io.PrintWriter;

import java.sql.ResultSet;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;


import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class SearchServlet extends HttpServlet
{
	public void doGet(HttpServletRequest req,HttpServletResponse res)throws
ServletException,IOException
	{
		try
		{
			res.setContentType("text/html");
			PrintWriter out=res.getWriter();
			String sid=req.getParameter("sid");
			StudentDao sd=new StudentDaoImpl();
			ResultSet rs=sd.search(sid);
			boolean b=rs.next();
			if(b==true)
			{
				out.println("<html>");
				out.println("<body bgcolor='lightyellow'>");
				out.println("<b><font size='6'><br>");
				out.println("<pre>");
				out.println("		Student Id....."+rs.getString(1));
				out.println();
				out.println("		Student Name......"+rs.getString(2));
				out.println();
				out.println("		Student Marks....."+rs.getInt(3));
				out.println("</pre></font></b></body></html>");
			}
			else
			{
				req.getRequestDispatcher("notexisted.html").forward(req,res);
			}
		}
		catch(Exception e)
		{
			e.printStackTrace();
		}
	}
}
```

EditFormServlet:

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.ResultSet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


public class EditFormServlet extends HttpServlet {
        public void doGet(HttpServletRequest req, HttpServletResponse res)
                        throws ServletException, IOException {
                try {
                        res.setContentType("text/html");
                        PrintWriter out = res.getWriter();
                        String sid = req.getParameter("sid");
                        StudentDao sd = new StudentDaoImpl();
                        ResultSet rs = sd.getStudent(sid);
                        boolean b = rs.next();
                        if (b == true) {
                                out.println("<html>");
                                out.println("<body bgcolor='lightgreen'>");
                                out.println("<b><font size='7'>");
                                out.println("<br>");
                                out.println("<form method='get' action='./update'>");
                                out.println("<pre>");
                                out.println("          Student Id            " + rs.getString(1));
                                out.println("<input type='hidden' name='sid' value='" + sid
                                                + "'/>");
                                out.println("          Student Name               <input
type='text' name='sname' value='"
                                                + rs.getString(2) + "'/>");
                                out.println();
                                out.println("          Student marks              <input
type='text' name='smarks' value='"
                                                + rs.getInt(3) + "'/>");
                                out.println();
                                out.println("          <input type='submit' value='Update'/>");
                                out.println("</pre></form></font></b></body></html>");
                        } else {
                                req.getRequestDispatcher("notexisted.html").forward(req, res);
                        }
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

UpdateServlet.java:

```
import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class UpdateServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse res)
            throws ServletException, IOException {


        try {
            String sid = req.getParameter("sid");
            String sname = req.getParameter("sname");
            int smarks = Integer.parseInt(req.getParameter("smarks"));
            StudentDao sd = new StudentDaoImpl();
            String status = sd.update(sid, sname, smarks);
            if (status.equals("success")) {
                req.getRequestDispatcher("success.html").forward(req, res);
            }
            if (status.equals("failure")) {
                req.getRequestDispatcher("failure.html").forward(req, res);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

DeleteServlet.java:

```
import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class DeleteServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse res)
            throws ServletException, IOException {
        String sid = req.getParameter("sid");
        StudentDao sd = new StudentDaoImpl();
        String status = sd.delete(sid);
        if (status.equals("success")) {
            req.getRequestDispatcher("success.html").forward(req, res);
        }
        if (status.equals("failure")) {
            req.getRequestDispatcher("failure.html").forward(req, res);
```

```
                }
                if (status.equals("notexisted")) {
                        req.getRequestDispatcher("notexisted.html").forward(req, res);
                }
        }
}
```

DURGASOFT Means JAVA
JAVA Means DURGASOFT
India's No.1 JAVA Trainer
More Than 1000 Students in a Single Class Room.
Weapon of DURGASOFT
Mr.Nagoor Babu M.Tech
Trained Lakhs of Students for Last 13years (Realtime Expert & Java Certified)

3. Applet-Servlet Communication:

In general in web application, we will use a browser as client, we will send request from client browser to a servlet available at server , by executing the respective servlet some response will be send back to the client browser.

Similarly in case of Applet-Servlet Communication, we will use applet as client, from the applet only we will send request to the respective servlet available at server machine, where by executing the respective servlet the required response will be generated and send back to the applet.

In above situation, the communication which we provided between applet and servlet is called as **Applet-Servlet Communication**.

If we want to achieve Applet-Servlet Communication in web applications we have to use the following steps.

Step 1: Prepare URL object with the respective url.

```
URL u=new

          URL(“http://localhost:8080/loginapp/login?uname=abc&upwd=abc123”);
```

Step 2: Establish connection between applet and server by using URLConnection object.

```
   URLConnection uc=u.openConnection();
```

Step 3: Send request from applet to servlet.

```
       uc.setDoInput(true);

      uc.setDoOutpput(true);
```

Note: If we do the above step a request will be send to servlet from applet where container will execute the respective servlet, generate the response and send that response to applet client. But, the response is available on URLConnection object.

Step 4: Get InputStream from URLConnection.

```
      InputSream is=uc.getInputStream();
```

Step 5: Read the response from InputStream.

```
     BufferedReader br=new BufferedReader(new InputStreamReader(is));

     String res=br.readLine();
```

loginapp1:-

web.xml:-

```
<web-app>
<servlet>
<servlet-name>LoginServlet</servlet-name>
<servlet-class>LoginServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>LoginServlet</servlet-name>
<url-pattern>/login</url-pattern>
</servlet-mapping>
</web-app>
```

LoginServlet.java:-

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class LoginServlet extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                PrintWriter out=response.getWriter();
                String uname=request.getParameter("uname");
                String upwd=request.getParameter("upwd");
                if(uname.equals("durga") && upwd.equals("durga")){
                        out.println("Login Success");
                }
                else{
                        out.println("Login Failure");
                }
        }

}
```

LoginApplet.java:-

```
import java.applet.Applet;
import java.awt.Button;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Label;
import java.awt.TextField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;

public class LoginApplet extends Applet implements ActionListener {
        Label l1,l2;
        TextField tf1,tf2;
        Button b;
        String res="";
        public void init(){
                this.setBackground(Color.pink);
                this.setLayout(new FlowLayout());
                l1=new Label("User Name");
                l2=new Label("Password");
                tf1=new TextField(20);
                tf2=new TextField(20);
                tf2.setEchoChar('*');
                b=new Button("Login");
                b.addActionListener(this);
                this.add(l1);
                this.add(tf1);
                this.add(l2);
                this.add(tf2);
                this.add(b);
        }
        public void actionPerformed(ActionEvent ae) {
                try{
                        URL u=new
URL("http://localhost:2011/loginapp1/login?uname="+tf1.getText()+"&upwd="+tf2.getTex
t());
                        URLConnection uc=u.openConnection();
                        uc.setDoInput(true);
                        InputStream is=uc.getInputStream();
                        BufferedReader br=new BufferedReader(new InputStreamReader(is));
                        res=br.readLine();
                        repaint();
                }catch (Exception e) {
                        e.printStackTrace();
                }
        }
        public void paint(Graphics g){
                Font f=new Font("arial", Font.BOLD, 30);
                g.setFont(f);
                g.drawString("Status :"+res, 50, 250);
        }
}
```

LoginApplet.html:-

```
<applet code="LoginApplet" width="500" height="500"></applet>
```

GlassFish Server:

GlassFish Server is an Application Server, provided by Sun Micro Systems.

GlassFish server will provide almost all the middle ware services what application servers are provided in general like JNDI, JMS, Java Mail, Security, JTA and so on.

GlassFish version3 is compatible with java6 and above, it is able to provide support support for servlet3.0, jsp2.1 and so on.

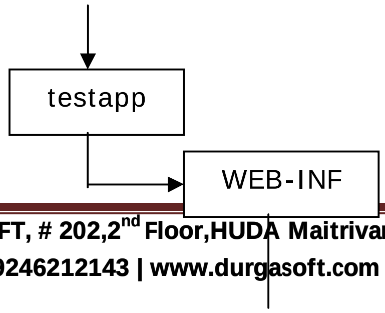
To design and execute servlets in GlassFish server we are able to use Tomcat provided servlet API implementation i.e. Servlet API.jar.

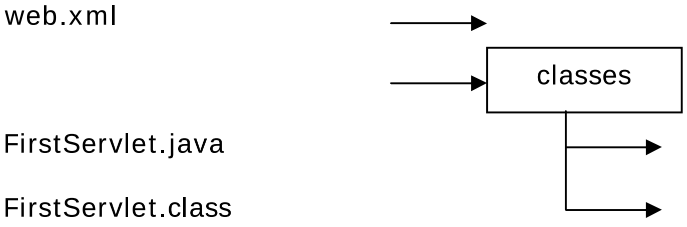
If we want to design and execute web applications in GlassFish server

we have to use the following steps.

Step 1: Prepare web application and its war file separately.

D:\apps




web.xml

classes

FirstServlet.java

FirstServlet.class

To compile all the servlet files GlassFish server has provided Servlet API implementation in the form of javax.servlet.jar file.

GlassFish server has provided javax.servlet.jar file at the following location.

```
 C:\glassfishv3\glassfish\modules

 D:\apps\testapp\WEB-INF\classes>set classpath=C:\glassfishv3\
     glassfish\modules\ javax.servlet.jar;

 D:\apps\testapp\WEB-INF\classes>javac *.java

 D:\apps\testapp>jar –cvf testapp.war *.*
```

Step 2: Start the Application server.

To start Glassfish server we have execute startserv.bat file provided by Glassfish server at the following location.

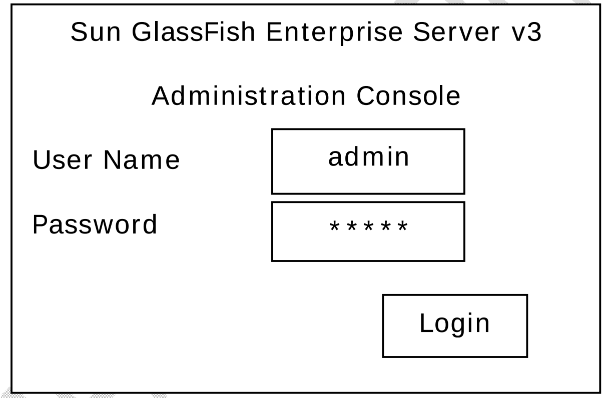
C:\glassfishv3\glassfish\bin

Step 3: Open Administration Console.

To open Administration Console in GlassFish server we have to use the following url at client browser.

http://localhost:4848

If we do the we above then Administration Console will be open, where we have to provide username and password.



Sun GlassFish Enterprise Server v3

Administration Console

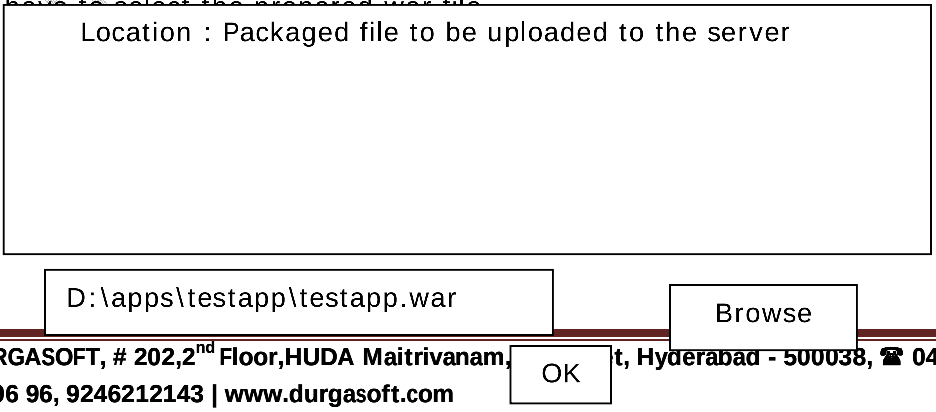
User Name: admin

Password: *****

Step 4: Deploy web application on GlassFish server.

If we click on Login button in Administration Console then automatically GlassFish server Home page will be open, where to deploy wed application go for Deployment section, where click on either List Deployed Application or Deploy an application.

If we click on Deploy an application automatically a browsing window will be open, where we have to select the prepared war file.



Location : Packaged file to be uploaded to the server

D:\apps\testapp\testapp.war

If we click on OK button, the specified war file will be uploaded to the server and the application context name will be displayed in the list of deployed applications.

NameEnabledEnginesAction

testapp ✓ web [Launch] [Redeploy] [Restart]

To access the web application we have to click on Launch hyper link under Action part.

If we click on Launch hyper link automatically a window will be open with the following url.

http://localhost:1111/testapp

If we write url pattern of the particular servlet then access the application

By using the following url.

http://localhost:1111/testapp/first

DURGASOFT Means JAVA
JAVA Means DURGASOFT
India's No.1 JAVA Trainer
More Than 1000 Students in a Single Class Room.
Weapon of DURGASOFT
Mr.Nagoor Babu M.Tech
Trained Lakhs of Students for Last 13years (Realtime Expert & Java Certified)