Experiment No. 7

Aim of the Experiment:

A. Write Servlet (procedure for client side) to display the username and password accepted from the client.

B. Write Servlet (procedure for server side) to display the username and password accepted from the client

Objective:

To understand client and server side servlet.

Course Outcome Addressed: CO6

Theory:

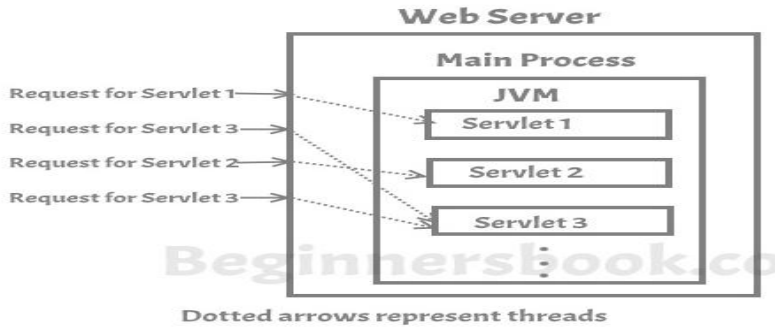
Servlet is a java program that runs inside JVM on the web server. It is used for developing dynamic web applications. The main **difference between static and dynamic web page** is that static page as name suggests remains same for all users however a dynamic web page changes based on the request from client (user's browser). To create servlets we need Servlet API. There are two packages that you must remember while using API, the javax.servlet package that contains the classes to support generic servlet (protocol-independent servlet) and the javax.servlet.http package that contains classes to support http servlet.

hierarchy of packages:

```
java.lang.Object
        |_extended byjavax.servlet.GenericServlet
        |_extended byjavax.servlet.http.HttpServlet
```

Every Servlet must implement the java.servlet.Servlet interface, you can do it by extending one of the following two classes: javax.servlet.GenericServlet or javax.servlet.http.HttpServlet. The first one is for protocol independent Servlet and the second one for http Servlet.

How servlet works?

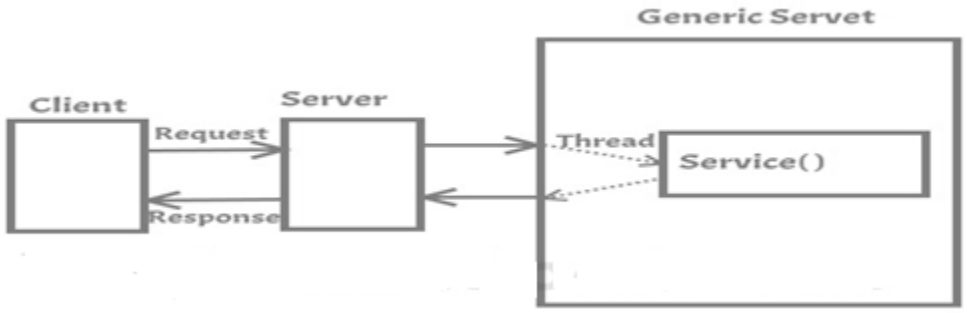


Generic Servlet

To create a Generic Servlet we must extend `javax.servlet.GenericServlet` class. GenericServlet class has an abstract service() method. Which means the subclass of GenericServlet should always override the service() method.
Signature of service() method:

```
public abstract void service(ServletRequest request, ServletResponse response)
        throws ServletException, java.io.IOException
```

The service() method accepts two arguments ServletRequest object and ServletResponse object. The request object tells the servlet about the request made by client while the response object is used to return a response back to the client.



HTTP Servlet

To create Http Servlet we must extend javax.servlet.http.HttpServlet class, which is an abstract class. Unlike Generic Servlet, the HTTP Servlet doesn't override the service() method. Instead it overrides one or more of the following methods. It must override at least one method from the list below:

- **doGet()** – This method is called by servlet service method to handle the HTTP GET request from client. The Get method is used for getting information from the server
- **doPost()** – Used for posting information to the Server
- **doPut()** – This method is similar to doPost method but unlike doPost method where we send information to the server, this method sends file to the server, this is similar to the FTP operation from client to server
- **doDelete()** – allows a client to delete a document, webpage or information from the server
- **init() and destroy()** – Used for managing resources that are held for the life of the servlet
- **getServletInfo()** – Returns information about the servlet, such as author, version, and copyright.

In Http Servlet there is no need to override the service() method as this method dispatches the Http Requests to the correct method handler, for example if it receives HTTP GET Request it dispatches the request to the doGet() method.

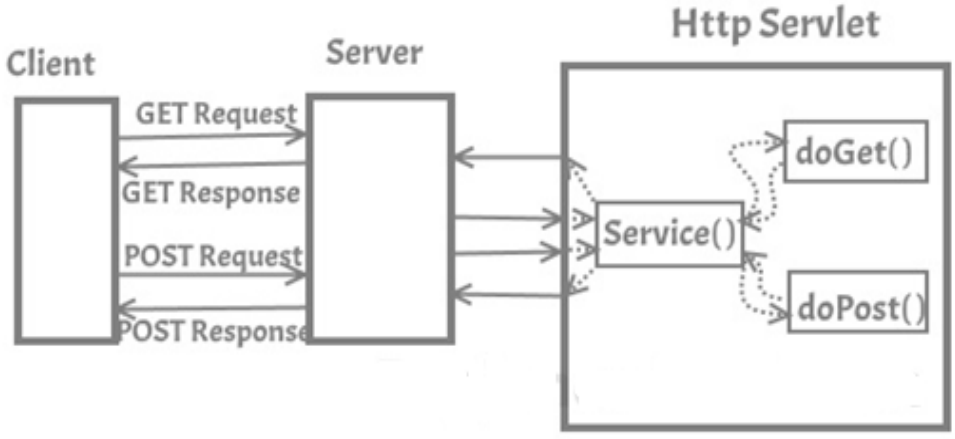
How Servlet Works?

1) When the web server (e.g. Apache Tomcat) starts up, the servlet container deploy and loads all the servlets. During this step Servlet container creates ServletContext object. **ServletContext is an interface that defines the set of methods that a servlet can use to communicate with the servlet container**.

Note: **There is only one ServletContext per webapp** which is common to all the servlets. ServletContext has several useful methods such as addListener(), addFilter() etc. For now I am not explaining them as I will cover them in a separate text about ServletContext.

2) Once the servlet is loaded, the servlet container creates the instance of servlet class. For each instantiated servlet, its init() method is invoked.

3) Client (user browser) sends an Http request to web server on a certain port. Each time the web server receives a request, the servlet container creates HttpServletRequest and HttpServletResponse objects. The HttpServletRequest object provides the access to the request information and the HttpServletResponse object allows us to format and change the http response before sending it to the client.
The servlet container spawns a new thread that calls service() method for each client request. **The service() method dispatches the request to the correct handler method based on the type of request**.
For example if server receives a Get Request the service() method would dispatch the request to the doGet() method by calling the doGet() method with request parameters. Similarly the requests like Post, Head, Put etc. are dispatched to the corresponding handlers doPost(), doHead(), doPut() etc. by service() method of servlet.



4) When servlet container shuts down, it unloads all the servlets and calls destroy() method for each initialized servlets.

Program:

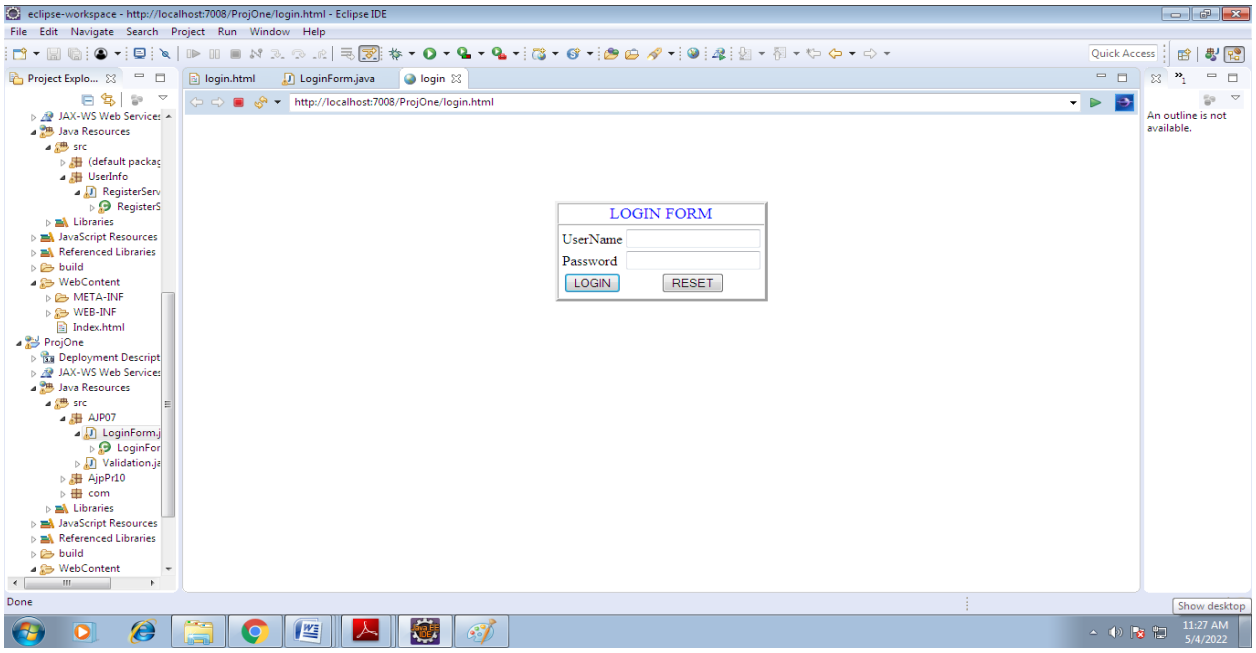
Login.html:

```
<html>
<head><title>login</title></head>
<body>
<form name="login form" method="post"
action="http://localhost:8080/examples/servlet/Validation">
<br/><br/><br/><br/><br/>
<table align="center" border="3" border color="blue" cellspacing="0"height="120">
<tr><td align="center"><font color="blue" size="4">LOGIN FORM</font></td></tr>
<tr><td><table><tr><td>UserName</td><td><input type="text" name="user"/></td></tr>
<tr><td>Password</td><td><input type="password" name="pwd"/></td></tr>
<tr><td align="center"><input type="submit" value="LOGIN"/></td><td
align="center"><input type="Reset" value="RESET"/></td></tr>
</table></td></tr></table></form></body>
</html>
```

Validation.java:

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
public class Validation extends GenericServlet
{
public void service(ServletRequest req,ServletResponse res)throws
ServletException,IOException
{
PrintWriter pw=res.getWriter();
String x=req.getParameter("user");
String y=req.getParameter("pwd");
if(x.equals("admin")&&y.equals("admin"))
pw.println("<font color='green' size='5'>Welcome to this webpage</font>");
else
pw.println("<font color='red' size='5'>Invalid username or password</font>");
pw.close();
}
}
```

Output:



Conclusion:

References:

Herbert Schildt , "Java : The Complete Reference" Tata McGraw-Hill (7th Edition).

Questions:

1. Explain the Servlet API.
2. How can a servlet get the name of the server and the port number for a particular request?
3. How can a servlet get information about the client machine?
4. What are the three methods of inter-servlet communication?