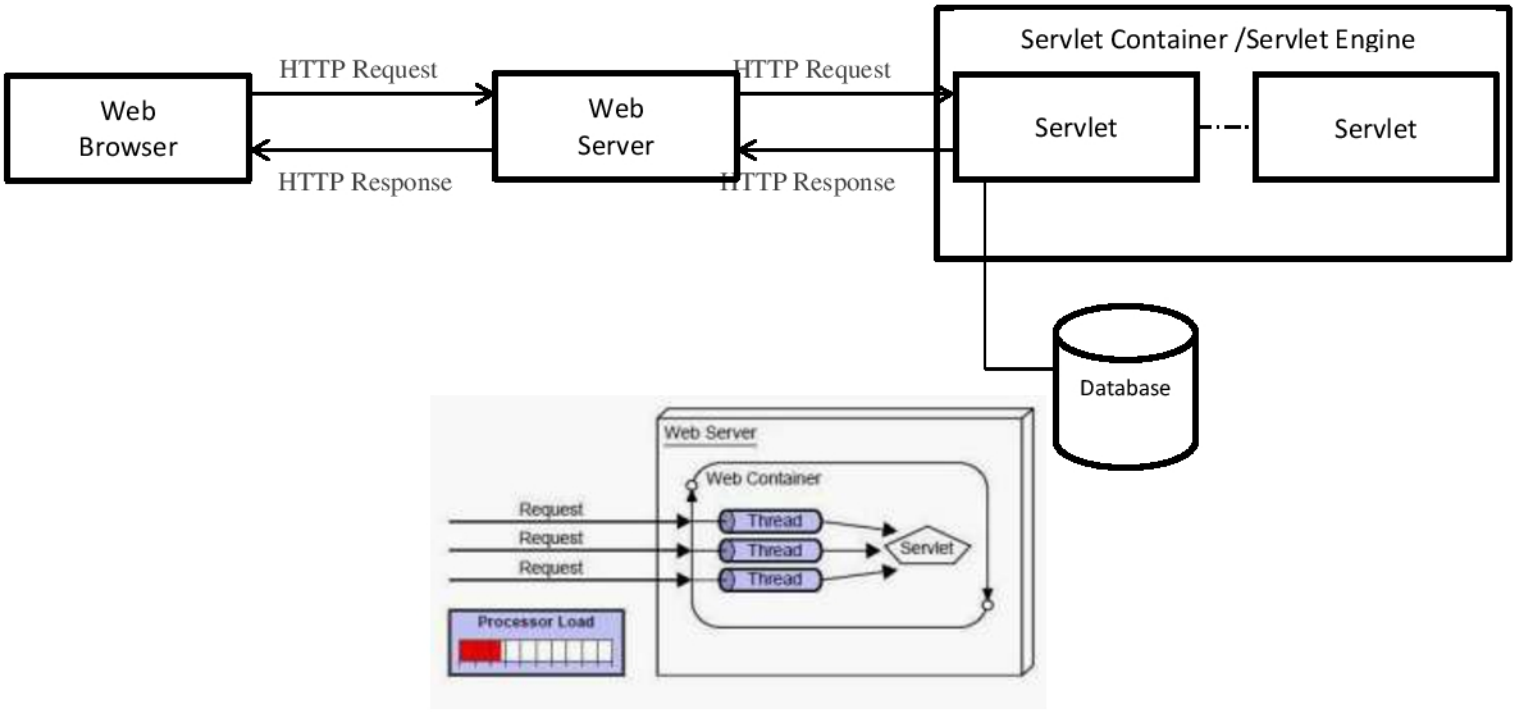
Chapter-6.1
Web Programming Using Java Servlet APIs

Java Servlets

- Java Servlets are server-side Java program modules that process and answer client requests and implement the servlet interface.
- It helps in enhancing Web server functionality with minimal overhead, maintenance and support by using HTTP protocol.
- Two packages **javax.servlet and javax.servlet.http** contains the classes and interfaces that are required for building servlet.
- **These are not part of java core packages hence are not included in JDK.**

The architecture of HTTP request response model for servlet is shown below.



- A servlet container is a server that executes a servlet and acts as an intermediary between the client and the server.
- As servlet modules run on the server, they can receive and respond to requests made by the client.
- Request and response objects of the servlet offer a convenient way to handle HTTP requests and send text data back to the client.
- **Servlet technology was the original Java based solution for web development however due to the problems of maintaining the HTML within the Java code they were never a great success. JSP was the solution to this.**

(Note: A container like Apache Tomcat, manages the runtime environment for servlet)

Working Principle

1. A client application sends HTTP request to the server.
2. The servlet container receives the request and directs it to be processed by the appropriate servlet. This container is also responsible for managing the lifecycle of a Servlet.
3. The servlet does its processing.
4. The servlet returns its result to the client normally in the form of an HTML document which can be displayed in a browser.

Life Cycle of servlet

- Servlet containers (example, Apache Tomcat Server) are responsible for handling requests, passing that request to the Servlet and then returning the response to the client.
- The basic lifecycle of a Servlet consists of three methods init(), service() and destroy() as explained below.

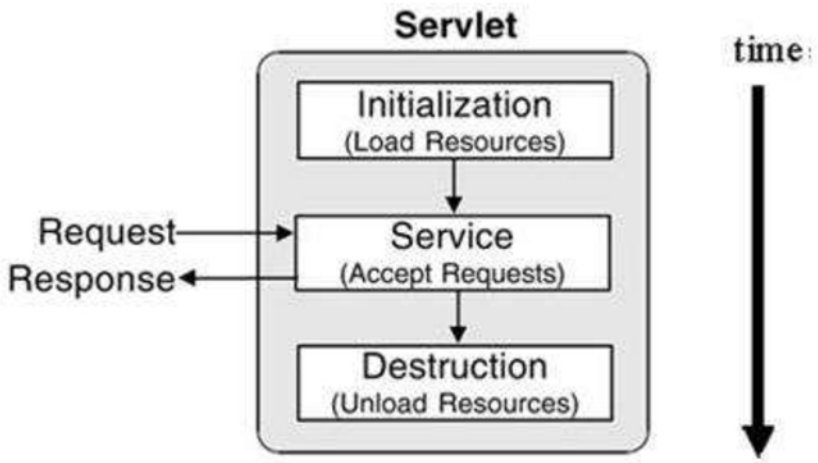


Fig: Servlet Lifecycle

- The Servlet Container creates an instance of the Servlet.
- The Container calls the instance's init() method.
- If the Container has a request for the Servlet it will call the instance's service() method.
- Before the instance is destroyed the Container will call the destroy() method.
- Finally, the instance is destroyed and marked for garbage collection

- Typically the init() and destroy() method is called only once and then the service() method is called repeatedly for each request. This is much more efficient than executing init(), service(), destroy() for each request.
- What happens, when a service () method is still executing when the Container receives another request ? Typically, this will involve the creation of another program execution thread. In practice, Servlet Containers create a pool of threads to which incoming requests are generally allocated.

The Servlet API

To build servlets, two package are required

1. javax.servlet package (Generic Servlet Class)
2. javax.servlet.http package (Http Servlet Class)

These packages are not part of the java core packages, hence are not included in Java development kit(JDK).

Installing apache Tomacat web server

Will be installed in LAB……

Steps in Working with Generic servlets using Tomcat web server

(A simple servlet program to display hello world)

1. Write servlet program extending javax.servlet.GenericServlet class using any texteditor.

```
import javax.servlet.*;
import java.io.*;
public class ServletDemo extends GenericServlet
{
        public void service(ServletRequest rq, ServletResponse rsp) throws ServletException, IOException
        {
```

```
                PrintWriter pw=rsp.getWriter();
                rsp.setContentType("text/html");
                pw.println("<html><body>Hello World</body></html>");
                pw.close();
        }
}
```

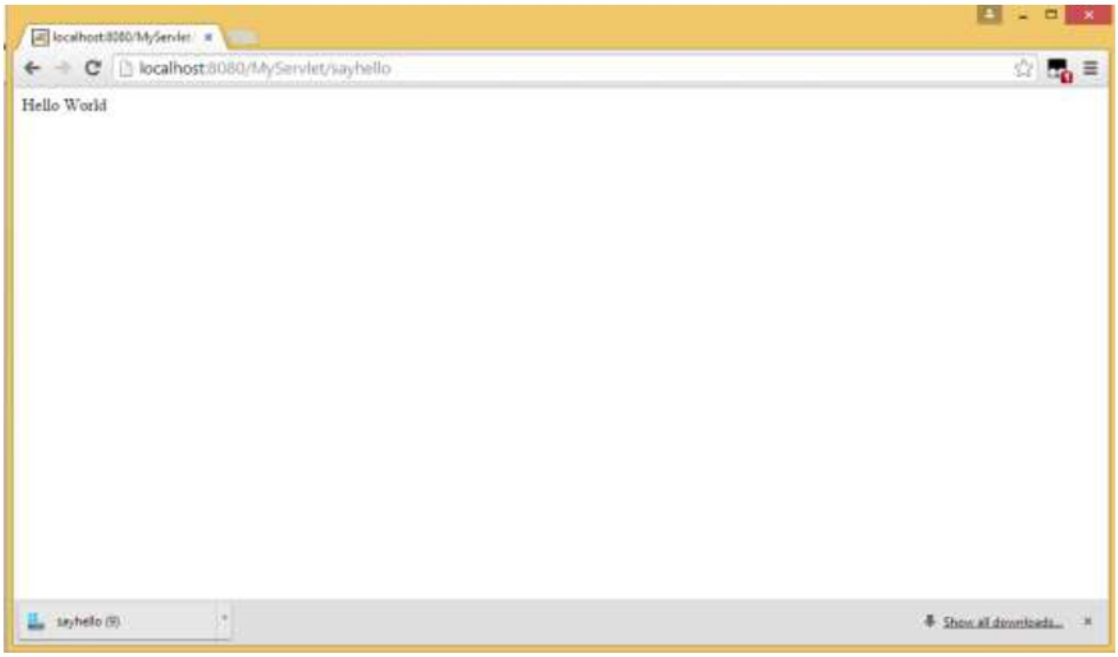
2. Create the following directory structure
   C:\apache-tomcat-8.0.14\webapps\MyServlet\WEB-INF\classes
3. Save the file with .java extension i.e. ServletDemo.java in classes sub directory.
4. Now compile using javac tool as below.

```
C:\apache-tomcat-8.0.14\webapps\MyServlet\WEB-INF\classes>javac -cp C:\apache-to
mcat-8.0.14\lib\servlet-api.jar ServletDemo.java
```

   (cp means class path, in exam you can just write **javac servlet-api.jar ServletDemo.java**)
5. This will create ServletDemo.class in **classes** sub directory.
6. Now create web.xml file. The xml file also called as **deployment descriptor** specifies various configuration parameters such as the name used to invoke the servlet(i.e. alias), a description of the servlet, the servlet fully qualified class name and a servlet mapping i.e. the path that causes the servlet container to invoke the servlet.

```
<?xml version="1.0"?>
<web-app>
        <servlet>
                <servlet-name>example</servlet-name>
                <servlet-class>ServletDemo</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>example</servlet-name>
                <url-pattern>/sayhello</url-pattern>
        </servlet-mapping>
</web-app>
```

7. Save this xml file in WEB-INF sub directory
   C:\apache-tomcat-8.0.14\webapps\MyServlet\WEB-INF
8. Run the servlet in the browser by calling the servlet in their context path by URL pattern specified in the web.xml file.
9. We will see the following output in the browser



Note:

1. Service() method handles requests from client. The first argument is ServletRequest object which enable to read data that is provided via the client request. The ServletRequest Interface is used to handle client request to access a servlet. It provides the information of a servlet like content type, content length, parameter names and values etc. some of them are

public String getContentType( )	Returns the MIME type of the request body.
public String getParameter(String name)	Returns the value of a request parameter as a string.
public String getProtocol( )	Returns the name and version of the request protocol.
public int getRemotePort( )	Returns the Internet Protocol (IP) port of the client.
public String getServerName( )	Returns the host name of the server to which the request was sent.

2. The second argument is ServletResponse object which enables the servlet to formulate a response for the client. The ServletResponse interface defines an object to help a Servlet in sending a response to the client. It has various methods that help a servlet to respond to the client requests. Some of them are

public void setContentType(String type)	Sets the content type of the response being sent to the client.
public PrintWriter getWriter( )	Returns the PrintWriter object that can be used to send character text to the client.

3. setContentType() method enable the browser to interpret the content as HTML source code.
4. getWritter() method obtains a PrintWriter. Anything written in this stream is sent to the client as a part of HTTP response.
5. println() method is used to write some simple HTML source code as the HTTP response.

Write servlet program to print your name 10 times

1. **Servelet Program**

```
import javax.servlet.*;
import java.io.*;
public class ServletDemo extends GenericServlet
{
        public void service(ServletRequest rq, ServletResponse rsp) throws IOException
        {
                PrintWriter pw=rsp.getWriter();
                for(int i=0;i<10;i++)
                {
                        pw.println("Bhesh Thapa");
                }
                pw.close();
        }
}
```

2 . XML file

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
        <servlet>
                <servlet-name>example</servlet-name>
                <servlet-class>ServletDemo</servlet-class>
        </servlet>

        <servlet-mapping>
                <servlet-name>example</servlet-name>
```

```
                <url-pattern>/showname</url-pattern>
        </servlet-mapping>
</web-app>
```

Write a servlet program to display the current system date time.

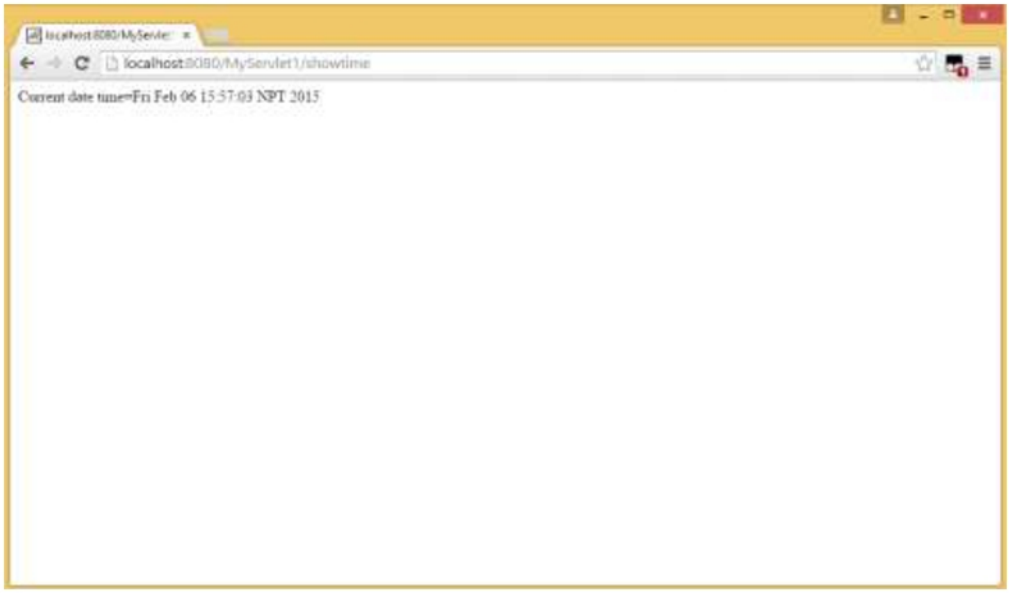
1. **Servlet Program**

```
import javax.servlet.*;
import java.io.*;
import java.util.*;
public class ServletDemo extends GenericServlet
{
        public void service(ServletRequest rq, ServletResponse rsp) throws  IOException
        {
                PrintWriter pw=rsp.getWriter();
                rsp.setContentType("text/html");
                Date dt=new Date();
                pw.println("<html>");
                pw.println("<body>");
                pw.println("Current date time="+dt);
                pw.println("</body>");
                pw.println("</html>");
                pw.close();
        }
}
```

2. **Web.xml**

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
        <servlet>
                <servlet-name>example</servlet-name>
                <servlet-class>ServletDemo</servlet-class>
        </servlet>

        <servlet-mapping>
                <servlet-name>example</servlet-name>
                <url-pattern>/showtime</url-pattern>
        </servlet-mapping>
</web-app>
```

Output:



Reading servlet parameters by extending GenericServlet Class

A servlet program that displays the name and roll number of a student sent by HTML file.

Servlet Program

```
import java.io.*;
import javax.servlet.*;
public class ServletParamDemo extends GenericServlet
{
        public void service(ServletRequest req, ServletResponse rsp) throws IOException
        {
                PrintWriter pw=rsp.getWriter();
                String strname=req.getParameter("txt_name");
                String strroll=req.getParameter("txt_roll");
                pw.write("Name="+strname);
                pw.write("  ");// for leaving some gap
                pw.write("Roll="+strroll);
                pw.close();
        }
}
```

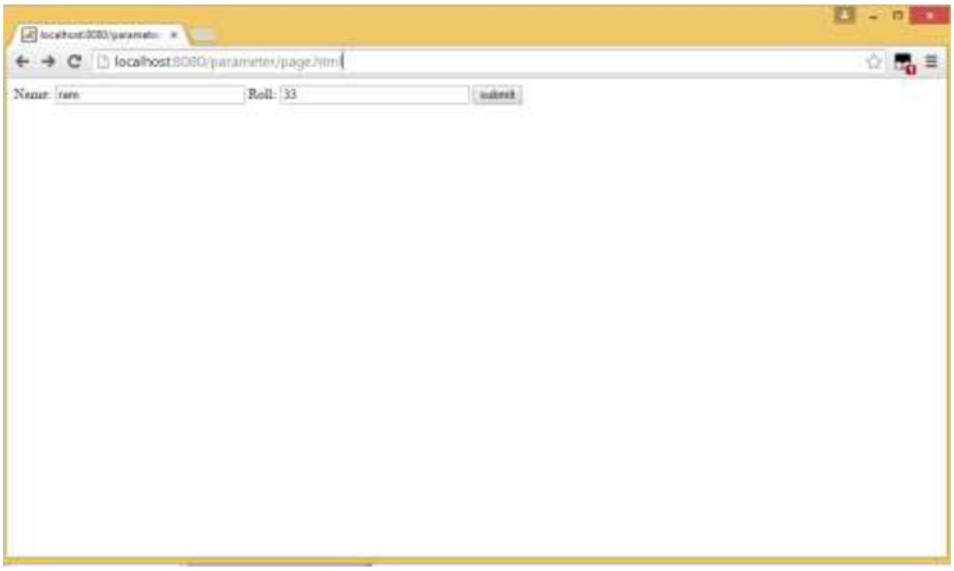
Web.xml

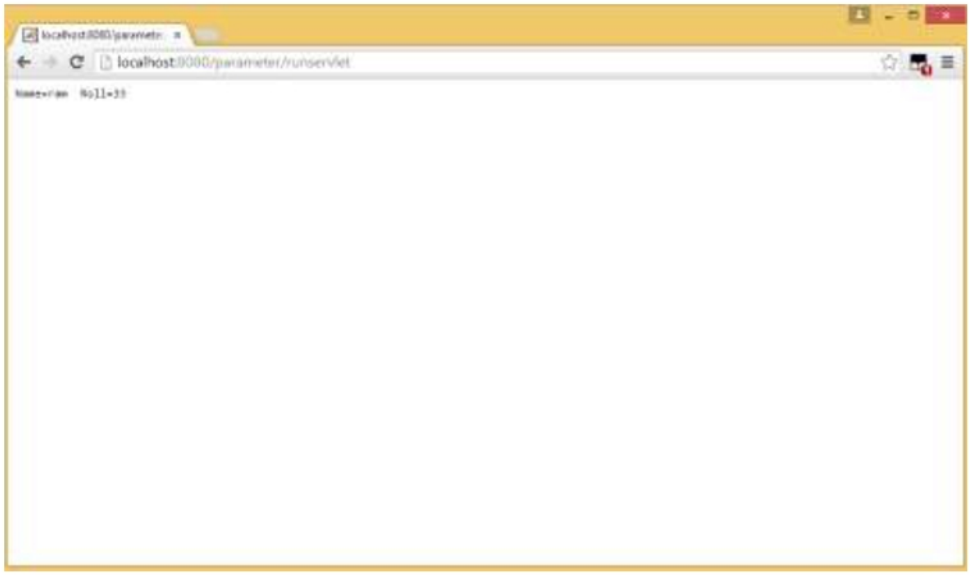
```
<?xml version="1.0" encoding ="ISO-8859-1"?>
        <web-app>
        <servlet>
                <servlet-name>example8</servlet-name>
                <servlet-class>ServletParamDemo</servlet-class>
        </servlet>

        <servlet-mapping>
                <servlet-name>example8</servlet-name>
                <url-pattern>/runservlet</url-pattern>
        </servlet-mapping>
</web-app>
```

HTML file

```
<html>
    <body>
        <form name="myfrm" method="post" action="runservlet">
        Name:
        <input type="textbox" name="txt_name" size=25 value="">
        Roll:
        <input type="textbox" name="txt_roll" size=25 value="">
        <input type="submit" value="submit">
        </form>
    </body>
</html>
```

Output





The javax.servlet.http package

- The javax.servlet.http package contains a number of interfaces and classes that can be used as APIs.
- HttpServletRequest, HttpServletResponse HttpSession etc are some of the interfaces provided by this package.
- HttpServletRequest enables servlet to read data from HTTP request.
- HttpServletresponse enables servlet to write data to an HTTP response.

Handling HTTP request and response

- We use doGet() and doPost() method for handling the GET and POST request which are automatically called by the HttpServlet's class's service method, which is called when a request arrives at the server. The service method performs the following task
  i. Determine the request type
  ii. Call the appropriate method i.e doGet() or doPost()
- The doGet() method is used for handling HTTP GET request.
- The doPost() method is used for handling HTTP POST request.
- The signature for doGet() and doPost() methods is same but the main difference is that doPost() is more secured than doGet() because we don't see the query parameters in the URL

Note: There is no need to call the service method in your servlet.

Get vs Post Method

GET	POST
Only limited amount of data can be sent because data is sent in header.	Large amount of data can be sent because data is sent in body.
Get request is not secured because query string appended in the URL bar.	Post request is secured because data is not exposed in the URL bar.
Get request can be bookmarked	Post request cannot be bookmarked.
A Get request is often cacheable.	A Post request can hardly cacheable.
Get request is more efficient and used more than post.	Post request is less efficient and used less than Get.

Creating a simple servlet using javax.servlet.http package

1. **Servlet Program**

```
import java.io.*;
import javax.servlet.http.*;
public class ServletDemo extends HttpServlet
{
        public void doGet(HttpServletRequest rq, HttpServletResponse rsp) throws  IOException
        {
                PrintWriter pw=rsp.getWriter();
                rsp.setContentType("text/html");
                pw.println("<html><body>Hello World</body></html>");
                pw.close();
        }
}
```

2. **Web.xml**

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
        <servlet>
                <servlet-name>example</servlet-name>
```

```
        <servlet-class>ServletDemo</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>example</servlet-name>
        <url-pattern>/sayhello</url-pattern>
    </servlet-mapping>
</web-app>
```

Passing parameter from HTML file to Servlet using doGet() Method

- Form is a feature of HTML that allows the user to input data through HTML documents.
- To create a form, the <FORM> tag is used.
- It is a non-empty tag and has three attributes associated with it. Name: Identifies name of the form Method: Can be specified GET (default) or POST (preferred)

  GET – attaches the input to the ACTION URL

  POST – sends the input in a data base separately

  Action : Specifies the path of the CGI script used to process the form.

Elements of Form:

1. **INPUT TAG:**
   - The <INPUT> tag is used to specify which input fields are available in the form.
   - <INPUT> tag is an empty tag with three attributes.
   - TYPE: This is set to TEXT, indication a single test input fields. There are other types like RADIO for radio button, CHECKBOX for checkbox, BUTTON for button etc.
   - NAME: This is a variable name for the text field that the author must specify.
   - SIZE: This is the width or the TEXT field.

     The various TYPE value available for the <INPUT> tag:

TEXT	:	Specifies a text field.
PASSWORD	:	Specifies the password field
CHECKBOX	:	Specifies the element checked
RADIO	:	Specifies a single toggle ON or OFF
FILE	:	Specifies the field for Browse
SUBMIT	:	Uploads the form to the server
RESET	:	Resets form fields to defaults

2. **SELECT TAG:**
   - The <SELECT> tag used for creating lists and formatting the text fields.
   - There are four attributes associated with the <SELECT> tag.

MULTIPLE	:	Indicates the number of elements in a list that can be displayed.
SIZE	:	Determines the number if items to be displayed in a list
OPTION	:	Defines each value within <SELECT> tag
NAME	:	Indicates the name of the field.

3. **TEXTAREA TAG:**
   - The <TEXTAREA> tag is used as a text entry field with multiple rows.
   - There are three attributes associated with the <TEXTAREA> tag.

ROWS	:	Specifies the height of the text field in characters.
COLS	:	Specifies the width of the text field in characters.
NAME	:	Specifies the name of the field.

- The following program demonstrates how to pass the HTML parameter into servlet using doget() method.

1. **Servlet program**

```
import java.io.*;
import javax.servlet.http.*;
public class ServletParamDemo extends HttpServlet
{
	public void doGet(HttpServletRequest req, HttpServletResponse rsp) throws IOException
	{
		PrintWriter pw=rsp.getWriter();
		String strname=req.getParameter("txt_name");
		String strroll=req.getParameter("txt_roll");
		String sem=req.getParameter("opt_sem");
		pw.write("Name="+strname);
		pw.write("  ");// for leaving some gap
		pw.write("Roll="+strroll);
		pw.write("  ");// for leaving some gap
		pw.write("Semester="+sem);
		pw.close();
	}

}
```

2. Web.xml

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
	<servlet>
		<servlet-name>example</servlet-name>
		<servlet-class>ServletParamDemo</servlet-class>
	</servlet>

	<servlet-mapping>
		<servlet-name>example</servlet-name>
		<url-pattern>/runservlet</url-pattern>
	</servlet-mapping>
</web-app>
```

3. Html file

```
<html>
	<body>
		<form name="myfrm" method="get" action="runservlet">
			Name:
			<input type="textbox" name="txt_name" size=25 value="">
			Roll:
			<input type="textbox" name="txt_roll" size=25 value="">
			Gender:
			<select name="opt_sem">
				<option value="Male">Male</option>
				<option value="Female">Female</option>
			</select>
			<input type="submit" value="submit">
		</form>
	</body>
```

```
</html>
```

Output:

Passing parameter from HTML file to Servlet using doPost() Method

1. **Servlet Program**

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class ServletParamDemo extends HttpServlet
{
	public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws ServletException, IOException
	{
		PrintWriter pw=rsp.getWriter();
		String strname=req.getParameter("txt_name");
		String strroll=req.getParameter("txt_roll");
		String sem=req.getParameter("opt_sem");
		pw.write("Name="+strname);
		pw.write("  ");// for leaving some gap
		pw.write("Roll="+strroll);
		pw.write("  ");// for leaving some gap
		pw.write("Semester="+sem);
		pw.close();
	}
}
```

2. **Web.xml**

```
	<?xml version="1.0" encoding ="ISO-8859-1"?>
	<web-app>
		<servlet>
			<servlet-name>example</servlet-name>
			<servlet-class>ServletParamDemo</servlet-class>
		</servlet>

		<servlet-mapping>
			<servlet-name>example</servlet-name>
			<url-pattern>/runservlet</url-pattern>
		</servlet-mapping>
	</web-app>
```

3. **Html file**

```
<html>
    <body>
    <form name="myfrm" method="post" action="runservlet">
        Name:
        <input type=textbox name="txt_name" size=25 value="">
        Roll:
        <input type=textbox name="txt_roll" size=25 value="">
        Semester:
        <select name="opt_sem">
            <option value="Male">Male</option>
            <option value="Female">Female</option>
        </select>
        <input type=submit value="submit">
        </form>
    </body>
</html>
```

Output:

Write a servlet program to display "welcome to our site" on clicking submit button if the user input is "Bhesh" for username field and "javaiseasy" for passwordfield.

1. **Servlet program**

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class ServletParamDemo extends HttpServlet
{
    public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws ServletException, IOException
    {
        PrintWriter pw=rsp.getWriter();
        String uname=req.getParameter("txt_uname");
        String pwd=req.getParameter("txt_pwd");
        if(uname.equals("Bhesh")&& pwd.equals("javaiseasy"))
        {
            pw.write("Welcome to our site");
        }
        else
        {
            pw.write("Invalid login");
        }
```

```
            pw.close();
        }

    }
```

2. **Web.xml**

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
    <servlet>
        <servlet-name>example</servlet-name>
        <servlet-class>ServletParamDemo</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>example</servlet-name>
        <url-pattern>/runservlet</url-pattern>
    </servlet-mapping>
</web-app>
```

3. **Html file**

```
`<html>
<body>
    <form name="myfrm" method="post" action="runservlet">
        Name:
        <input type="textbox" name="txt_uname" size=25 >
        Password:
        <input type="password" name="txt_pwd" size=25>
        <input type="submit" value="submit">
        </form>
    </body>
</html>
```

Output:

- **A database consists of table named tbl_login having fields id, uname and pwd. Write a servlet program to display suitable message in browser if user input valid username and password. (DSN: Bhesh)**

Note: To use type-4 driver, the connection is obtained as below

```
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
String connectiostring="jdbc:sqlserver://<Server Name>:<port>;user=<Username>;password=<Password>;databaseName=<database name>";
Connection conn = DriverManager.getConnection(connectiostring);
```

Servlet Program

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class LoginDemo extends HttpServlet
{
	public void doPost(HttpServletRequest req, HttpServletResponse rsp)  throws IOException
	{

		PrintWriter pw=rsp.getWriter();
		try
		{
			String uname=req.getParameter("txt_uname");
			 String pwd=req.getParameter("txt_pwd");

			 Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
			 String url = "jdbc:odbc:Bhesh";
			 Connection conn = DriverManager.getConnection(url,"","");
			 Statement stmt = conn.createStatement();
			 ResultSet rs;
			 rs = stmt.executeQuery("SELECT * FROM tbl_login where uname='"+uname+"' and pwd='"+pwd+"' ");
			 if(rs.next()==true)
			 {
				pw.write("Welcome to our site");
			 }

			 else
			 {
				pw.write("Invalid login");
			 }
			 pw.close();
		}
		catch (Exception f)
		{
			pw.write("Exception"+f);
		}
	}
}
```

Web.xml

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
	<servlet>
		<servlet-name>example2</servlet-name>
		<servlet-class>LoginDemo</servlet-class>
	</servlet>

	<servlet-mapping>
		<servlet-name>example2</servlet-name>
		<url-pattern>/runservlet</url-pattern>
	</servlet-mapping>
```

```
</web-app>
```

HTML file

```
<html>
    <body>
        <form name="myfrm" method="post" action="runservlet">
            Name:
            <input type="textbox" name="txt_uname" size=25 >
            Password:
            <input type="password" name="txt_pwd" size=25>
            <input type="submit" value="submit">
        </form>
    </body>
</html>
```

A database db_student consists of table tbl_student having fields id,name address and phone number.

i. Create a suitable user interface that allow users to input the data for a student. When the user press submit button pass these information to the Servlet for storing these data in the table tbl_student.[Assume DSN: Bhesh]

ii. Create a Servlet that connects to the database db_student and displays all the records stored in the table tbl_student.

Solution-i:

Html file

```
<html>
    <body>
        <form name="myform" method="post" action="runservlet">
            ID:
            <input type="text" name="id">
            <br/>Name:
            <input type="text" name="name">
            <br/>Address:
            <input type="text" name="add">
            <br/>Phone Number:
            <input type="text" name="phone">
            <br/>
            <input type="submit" value="Save">
        </form>
    </body>
</html>
```

Servlet Program

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class InsertDemo extends HttpServlet
{
    public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws IOException
    {
        String id=req.getParameter("id");
        String name=req.getParameter("name");
        String add=req.getParameter("add");
        String ph=req.getParameter("phone");
```

```
                PrintWriter pw=rsp.getWriter();
                try
                {
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                        String url = "jdbc:odbc:Bhesh";
                        Connection conn = DriverManager.getConnection(url,"","");
                        Statement st = conn.createStatement();
                        st.executeUpdate("insert into tbl_student values ('"+id+"','"+name+"','"+add+"','"+ph+"')");
                        pw.write("Inserted to database successfully");
                }
                catch (Exception e)
                {
                        pw.write("Error Occured"+e);
                }

        }

}
```

Web.xml

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
        <servlet>
                <servlet-name>example2</servlet-name>
                <servlet-class>InsertDemo</servlet-class>
        </servlet>

        <servlet-mapping>
                <servlet-name>example2</servlet-name>
                <url-pattern>/runservlet</url-pattern>
        </servlet-mapping>
</web-app>
```

Solution-ii:

Servlet Program

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class DisplayDemo extends HttpServlet
{
        public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws IOException
        {
                PrintWriter pw=rsp.getWriter();
                try
                {
                        rsp.setContentType("text/html");
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                        String url = "jdbc:odbc:Bhesh";
                        Connection conn = DriverManager.getConnection(url,"","");
                        Statement stmt = conn.createStatement();
                        ResultSet rs;
                        rs=stmt.executeQuery("Select * from tbl_student");
```

```
                    while (rs.next())
                    {
                            pw.write (rs.getString(1))
                            pw.write (" " )
                            pw.write (rs.getString(2))
                            pw.write (" " )
                            pw.write (rs.getString(3))
                            pw.write (" " )
                            pw.write (rs.getString(4))
                            pw.write("<br>");

                    }
                    rs.close();
                    stmt.close();
                    conn.close();
            }
            catch (Exception e)
            {
                    pw.write("Error Occured"+e);
            }

    }

}
```

Web.xml

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
    <servlet>
        <servlet-name>example2</servlet-name>
        <servlet-class>DisplayDemo</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>example2</servlet-name>
        <url-pattern>/runservlet</url-pattern>
    </servlet-mapping>
</web-app>
```

Create a user interface with two textfields to input the name and address, two radio buttons to select the gender: male/female, three checkbox to choose the hobby: singing, dancing, studying and a submit button. Write a servlet program to insert these information into the table named tbl_student (name, address, gender, hobby.comment). [Assume DSN: Bhesh].

Servlet Program:

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class ServletDemo extends HttpServlet
{
    public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws IOException
    {
        PrintWriter pw=rsp.getWriter();
```

```
                String fullname=req.getParameter("fname");
                String address=req.getParameter("add");
                String gender=req.getParameter("gender");
                String[] h = req.getParameterValues("hobby");
                String cmt=req.getParameter("txtcomment");

                String hoby="";
                 if (h != null )
                 {
                        int length = h.length;
                        for (int i=0; i<length; i++)
                        {
                                hoby=hoby+" " + h[i];
                        }
                 }

                try
                {
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                        String url = "jdbc:odbc:Mac";
                        Connection conn = DriverManager.getConnection(url,"","");
                        Statement st = conn.createStatement();
int i=st.executeUpdate("insert into tbl_student values ('"+fullname+"','"+address+"','"+gender+"','"+hoby+"','"+cmt+"')");
                        pw.write(i+" Record inserted");
                        conn.close();
                }
                catch (Exception e)
                {
                        pw.write("Error"+e);
                }
                pw.close();

        }
    }
```

HTML file

```
<html>
    <body>

        <form name=”myform” method="post" action="runservleta">
                First Name:
                <input type="text" name="fname">
                <br>
                Address:
                <input type="text" name="add">
                <br>
                Gender:
                <input type="radio" name="gender" value="Male"> Male
```

```
                    <input type="radio" name="gender" value="Female">Female
                    <br>
                    Hobbies:
                    <input type="checkbox" name="hobby" value="Music"> Music
                    <input type="checkbox" name="hobby" value="Reading"> Reading
                    <input type="checkbox" name="hobby" value="Cooking"> cooking
                    <br>
                    Comments:
                    <textarea cols=20 rows=4 name="txtcomment"></textarea>
                    <br>
                    <input type="submit" value="Submit">
            </form>
        </body>
</html>
```

Web.xml

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
<servlet>
        <servlet-name>examplea</servlet-name>
        <servlet-class>ServletDemo</servlet-class>
</servlet>

<servlet-mapping>
        <servlet-name>examplea</servlet-name>
        <url-pattern>/runservleta</url-pattern>
</servlet-mapping>
</web-app>
```

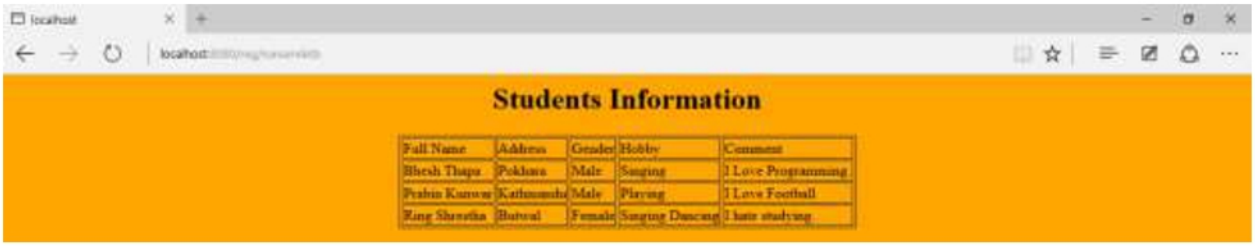
Write a servlet program to display all the records of students stored in the table tbl_student(Full Name, Address, Gender, hobby, comment) in tabular format when the user clicks the submit button.

Servlet Program:

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;

public class ServletDemo extends HttpServlet
{
        public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws  IOException
        {
                PrintWriter pw=rsp.getWriter();
                try
                {
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                        String url = "jdbc:odbc:Bhesh";
                        Connection conn = DriverManager.getConnection(url,"","");
                        Statement st = conn.createStatement();
                        ResultSet rs;
                        rs=st.executeQuery("Select * from tbl_student");
```

```
            rsp.setContentType("text/html");
            pw.write("<html>");
            pw.write("<body bgcolor=orange>");
            pw.write("<h1 align=center>Students Information</h1>");
            pw.write("<table align=center border="1">");
            pw.write("<tr>");
            pw.write("<td>");
            pw.write("Full Name");
            pw.write("</td>");
            pw.write("<td>");
            pw.write("Address");
            pw.write("</td>");
            pw.write("<td>");
            pw.write("Gender");
            pw.write("</td>");
            pw.write("<td>");
            pw.write("Hobby");
            pw.write("</td>");
            pw.write("<td>");
            pw.write("Comment");
            pw.write("</td>");
            pw.write("</tr>");
            while(rs.next())
            {
                  String fn=rs.getString(1);
                  String add=rs.getString(2);
                  String gen=rs.getString(3);
                  String hob=rs.getString(4);
                  String cmt=rs.getString(5);

                  pw.write("<tr>");
                  pw.write("<td>");
                  pw.write(fn);
                  pw.write("</td>");
                  pw.write("<td>");
                  pw.write(add);
                  pw.write("</td>");
                  pw.write("<td>");
                  pw.write(gen);
                  pw.write("</td>");
                  pw.write("<td>");
                  pw.write(hob);
                  pw.write("</td>");
                  pw.write("<td>");
                  pw.write(cmt);
                  pw.write("</td>");
                  pw.write("</tr>");
            }
            pw.write("</table>");
            pw.write("</body>");
            pw.write("</html>");
      }
```

```
            catch (Exception e)
            {
                  pw.write("Error"+e);
            }
            pw.close();
            conn.close();

      }
}
```



Cookies

- A cookie is a small text file that is stored in the client machine, which will be sent to the server on each request.
- A cookie has a name, a single value, and optional attributes such as a comment, path and domain qualifiers, a maximum age, and a version number.

How it work?

- By default, each request is considered as a new request.
- In cookies technique, we add cookie with response from the servlet.
- So cookie is stored in the cache of the browser.
- After that if request is sent by the user, cookie is added with request by default.
- Thus, we recognize the user as the old user.

Advantage of Cookies

- Simplest technique of maintaining the state.
- Cookies are maintained at client side.

Disadvantage of Cookies

- It will not work if cookie is disabled from the browser.
- Only textual information can be set in Cookie object.

- Following constructors are used to create cookie.

Constructor	Description
Cookie()	constructs a cookie.
Cookie(String name, String value)	constructs a cookie with a specified name and value.

- Useful Methods of cookie class

Method	Description
public void setMaxAge(int expiry)	Sets the maximum age of the cookie in seconds.
public String getName()	Returns the name of the cookie. The name cannot be changed after creation.
public String getValue()	Returns the value of the cookie.
public void setName(String name)	changes the name of the cookie.
public void setValue(String value)	changes the value of the cookie.

  Similarly other methods include, String getValue(), int getMaxAge() , void setComment(String), String getComment() etc.
- For adding cookie or getting the value from the cookie, we need some methods provided by other interfaces. They are:

1. public void addCookie(Cookie ck):method of HttpServletResponse interface is used to add cookie in response object.
2. public Cookie[] getCookies():method of HttpServletRequest interface is used to return all the cookies from the browser.

Example:

A servlet program that displays the last access time and current access time.

Note: the program assumes that only one cookie i.e. ck_date is to be read and written

1. **Servlet Program**

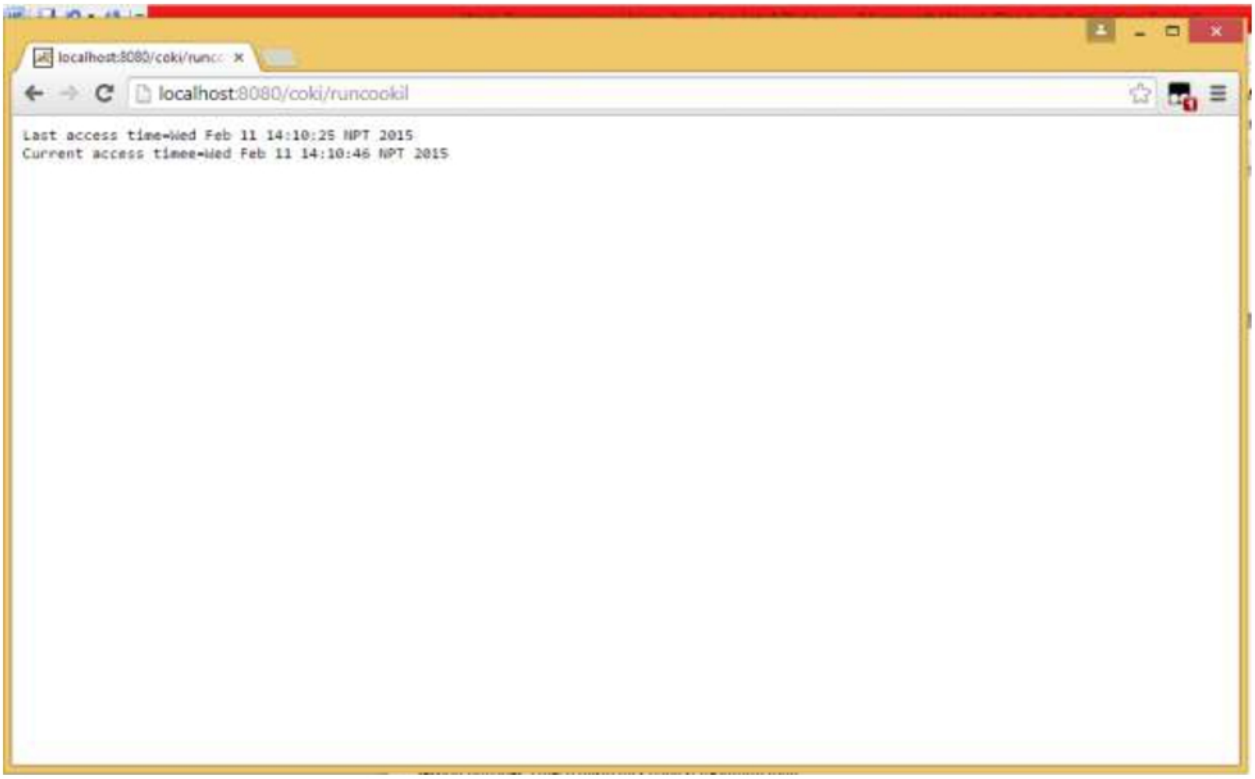
```
import java.io.*;
import java.util.Date;
import javax.servlet.http.*;
public class ServletDemo extends HttpServlet
{
        public void doGet(HttpServletRequest rq, HttpServletResponse rsp) throws IOException
        {
                String last_value="N/A";
                PrintWriter pw=rsp.getWriter();
                try
                {
                        Cookie [] ck=rq.getCookies();//get cookies from header of http
                        last_value=ck[0].getValue();// obtain the information from cookie stored in 0th index
                        //pw.write("Cookie Name="+ck[0].getName() );
                }
                catch(Exception e){}
                pw.write("Last access time="+last_value);
                pw.write(" ");
                Date dt=new Date();
                String dt_nospace=dt.toString().replaceAll(" ", ""); // using built in method
                Cookie ck_current=new Cookie("ck_date",dt_nospace);// create a cookie whose name is ck_date
                                                                                        //with value dt
                //Apache Tomcat8 and later version does'nt support White Space in Cookie but Earlier does.
                rsp.addCookie(ck_current);
                pw.write("\nCurrent access time="+dt.toString());
        }
}
```

2. **Web.xml**

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
<web-app>
        <servlet>
                <servlet-name>example</servlet-name>
                <servlet-class>ServletDemo</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>example</servlet-name>
                <url-pattern>/runcooki</url-pattern>
```

```
        </servlet-mapping>
    </web-app>
```

Output:



Create a HTML form with a Textfield Name for entering username and a submit button. Upon clicking the submit display welcome message and GO button in Servlet1. When user clicks the Go button use cookies to store and display another Hello message in second servlet.

HTML File: index.html

```
<form action="servlet11" method="post">
Name:<input type="text" name="userName"/><br/>
<input type="submit" value="go"/>
</form>
```

Servlet Program: FirstServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
    public class FirstServlet extends HttpServlet
     {
          public void doPost(HttpServletRequest request, HttpServletResponse response)
        {
            try
            {
                    response.setContentType("text/html");
                    PrintWriter out = response.getWriter();

                    String n=request.getParameter("userName");
                    out.print("Welcome "+n);

                    Cookie ck=new Cookie("uname",n);//creating cookie object
```

```
                                response.addCookie(ck);//adding cookie in the response

                                //creating submit button
                                out.print("<form method='post' action='servlet22'>");
                                out.print("<input type='submit' value='go'>");
                                out.print("</form>");
                                out.close();

                        }
                    catch(Exception e)
                    {
                            System.out.println(e);
                    }
                }
        }
```

Servlet Program: SecondServlet.java

```
        import java.io.*;
        import javax.servlet.*;
        import javax.servlet.http.*;

        public class SecondServlet extends HttpServlet
        {
        public void doPost(HttpServletRequest request, HttpServletResponse response)
        {
                try
                {
                        response.setContentType("text/html");
                        PrintWriter out = response.getWriter();

                        Cookie ck[]=request.getCookies();
                        out.print("Welcome Again "+ck[0].getValue());
                        out.close();
                }
                catch(Exception e)
                {
                        System.out.println(e);
                }
           }
        }
```

Web.xml

```
        <web-app>
        <servlet>
        <servlet-name>s11</servlet-name>
        <servlet-class>FirstServlet</servlet-class>
        </servlet>

        <servlet-mapping>
        <servlet-name>s11</servlet-name>
        <url-pattern>/servlet11</url-pattern>
```

```
</servlet-mapping>

<servlet>
<servlet-name>s22</servlet-name>
<servlet-class>SecondServlet</servlet-class>
</servlet>

<servlet-mapping>
<servlet-name>s22</servlet-name>
<url-pattern>/servlet22</url-pattern>
</servlet-mapping>

</web-app>
```

Session Tracking

- Session provides a mechanism for saving state information.
- A session can be created by geSesson() method of HttpServletRequest.
- The session state is shared among all the servlets that are associated with a particular client.
- Tomcat has a default timeout of 30 minutes but the default timeout depends on different web containers.
- The HttpServletRequest interface provides two methods to get the object of HttpSession:
  1. **public HttpSession getSession():**Returns the current session associated with this request, or if the request does not have a session, creates one.
  2. **public HttpSession getSession(boolean create):**Returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session.
  3. Commonly used methods of HttpSession interface
  1. **public String getId():**Returns a string containing the unique identifier value.
  2. **public long getCreationTime():**Returns the time when this session was created, measured in milliseconds since midnight January 1, 1970 GMT.
  3. **public long getLastAccessedTime():**Returns the last time the client sent a request associated with this session, as the number of milliseconds since midnight January 1, 1970 GMT.
  4. **public void invalidate():**Invalidates this session then unbinds any objects bound to it.
  5. Similarly, setAttribute(), getAttriubute() ,removeAttribute() etc. are some of the methods used frequently for session management.

Use

Provides a way to identify a user across more than one page request or visit to a website.

A servlet program that displays the last access time and current access time

1. **Servlet program**

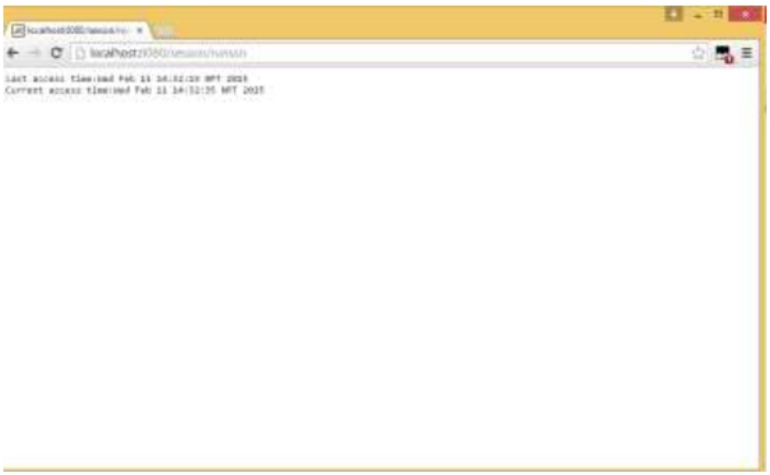
```
import java.io.*;
import java.util.Date;
import javax.servlet.http.*;
public class ServletDemo extends HttpServlet
{
        public void doGet(HttpServletRequest rq, HttpServletResponse rsp) throws IOException
        {
                HttpSession hs=rq.getSession(true); //get the HttpSesson object
                PrintWriter pw=rsp.getWriter();
```

```
                Date dt=(Date)hs.getAttribute("dat");
                if(dt!=null)
                {
                        pw.write("Last access time:"+dt);
                }
                dt=new Date();//get current date
                hs.setAttribute("dat",dt);   //Date is session name and dt is value to be stored in session dat
                pw.write("\nCurrent access time:"+dt);
        }
}
```

2. **Web.xml**

```
<?xml version="1.0" encoding ="ISO-8859-1"?>
    <web-app>
        <servlet>
            <servlet-name>exampleefle</servlet-name>
            <servlet-class>ServletDemo</servlet-class>
        </servlet>
        <servlet-mapping>
            <servlet-name>exampleefle</servlet-name>
            <url-pattern>/runssn</url-pattern>
        </servlet-mapping>
    </web-app>
```

Output:



Create a HTML form with a Textfield and Password field for entering username and password and a submit button. Write servlet program that validates if the username input "ram" and "java is easy" for the user name and password respectively. Redirect the request to another servlet to display welcome message "Welcome <username>" when the login was successful.

i. **Pass the value using concept of session**
ii. **Pass the value using the concept of Query String**

Solution-i

Html: index.html

```
<html>
	<body>
	<form name="myfrm" method="post" action="showlogin">
		Name:
		<input type="textbox" name="txt_uname" size=25 >
		Password:
		<input type="password" name="txt_pwd" size=25>
		<input type="submit" value="submit">
	</form>
	</body>
</html>
```

Servlet 1: LoginDemo.java

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class LoginDemo extends HttpServlet
{
	public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws IOException
	{
		PrintWriter pw=rsp.getWriter();
		try
		{
			String uname=req.getParameter("txt_uname");
			String pwd=req.getParameter("txt_pwd");
			if(uname.equals("ram") && pwd.equals("javaiseasy"))
			{
				HttpSession hs=req.getSession(true);
				hs.setAttribute("username",uname);
				rsp.sendRedirect("showwelcome");
			}
			else
			{
				pw.write("Invalid Login");

			}
		   pw.close();
		}
		catch (Exception f)
		{
			pw.write("Exception Occured: "+f);
		}
	}
}
```

Servlet 2: Wel.java

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class Wel extends HttpServlet
{
	public void doGet(HttpServletRequest req, HttpServletResponse rsp) throws IOException
	{
		PrintWriter pw=rsp.getWriter();
```

```
                try
                {
                        HttpSession hs= req.getSession(false);
                        String unam= (String)hs.getAttribute("username");//getAttribute Returns Object
                        pw.write("Welcome" + unam);
                }
                catch (Exception f)
                {
                        pw.write("Exception Occured: "+f);
                }
        }
}
```

Web.xml

```
<?xml version="1.0"?>
<web-app>
        <servlet>
                <servlet-name>serv1</servlet-name>
                <servlet-class>LoginDemo</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>serv1</servlet-name>
                <url-pattern>/showlogin</url-pattern>
        </servlet-mapping>

        <servlet>
                <servlet-name>serv2</servlet-name>
                <servlet-class>Wel</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>serv2</servlet-name>
                <url-pattern>/showwelcome</url-pattern>
        </servlet-mapping>
</web-app>
```

Solution-ii

Html file: index.html
Same as previous

Servlet1: LoginDemo.java

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class LoginDemo extends HttpServlet
{
        public void doPost(HttpServletRequest req, HttpServletResponse rsp) throws IOException
        {
                PrintWriter pw=rsp.getWriter();
                try
                {
                        String uname=req.getParameter("txt_uname");
```

```
                        String pwd=req.getParameter("txt_pwd");
                        if(uname.equals("ram") && pwd.equals("javaiseasy"))
                        {
                                rsp.sendRedirect("showwelcome?nm="+uname);

                        }
                        else
                        {
                                pw.write("Invalid Login");

                        }
                   pw.close();
                }
                catch (Exception f)
                {
                        pw.write("Exception Occured: "+f);
                }
        }
}
```

Servlet2: Wel.java

```
import java.io.*;
import javax.servlet.http.*;
import java.sql.*;
public class Wel extends HttpServlet
{
        public void doGet(HttpServletRequest req, HttpServletResponse rsp) throws IOException
        {
                PrintWriter pw=rsp.getWriter();
                try
                {
                        String unam= (String) req.getAttribute("nm");//getAttribute Returns Object
                        pw.write("Welcome" + unam);
                }
                catch (Exception f)
                {
                        pw.write("Exception Occured: "+f);
                }
        }
}
```

Web.xml
Same as previous

Assignment:

1. A HTML form consists of two textfields for entering full name and address, three checkbox for selecting hobbies: Music, Reading and Cooking, two radiobuttons for selecting gender:male,female and a textarea for entering comments. Write a servlet program to read and display these data from HTML form using Post() method when the user press submit button.
2. Write Short notes on
   i. Request dispatching – The forward() method, the include() method
   ii. Forward() vs sendRedirect()
   iii. Forward() vs include() method