Managing State

HTTP is a stateless (or non-persistent) protocol. Each request is treated by its own. A request will not know what was done in the previous requests. The protocol is designed to be stateless for simplicity. However, some Internet applications, such as e-commerce shopping cart, require the state information to be passed one request to the next. Since the protocol is stateless, it is the responsibility of the application to maintain state information within their application. Information about individual visit to a Web site is called state information. HTTP was originally designed to be stateless, which means that Web browser stored no data about pages viewed on previous visits to a Web site. A few techniques can be used to maintain state information across multiple HTTP requests, namely.

1. Field/Hidden fields of the HTML form
2. URL rewriting.
3. Sessions
4. Cookie

1) Field/Hidden fields of the HTML form

The example below shows a form file name is "InputForm1.html" with an input field and a Submit button. When a user submits the data by clicking on "Submit", the form data is sent to the File specified in the action attribute of the <form> tag.

Create a new document file in your text editor. Type the below script and save the document InputForm1.html file in the tomcat/webapps/ProjectName folder.

InputForm1.html

```
<!DOCTYPE html>
<html>
<body>
  <h1>Input Data Form </h1>
   <form method="post" action="SendData1.jsp">
      Name: <input type="text" name="txtname">
     FatherName: <input type="text" name="txtfname">
     <input type="submit" > <input type="reset" >
  </form>
</body>
</form>
```

After the save file we open InputForm1.html in web browser by entering the URL: - http://localhost:8080/InputForm1.html

Retrieving Submitted Data by post method:

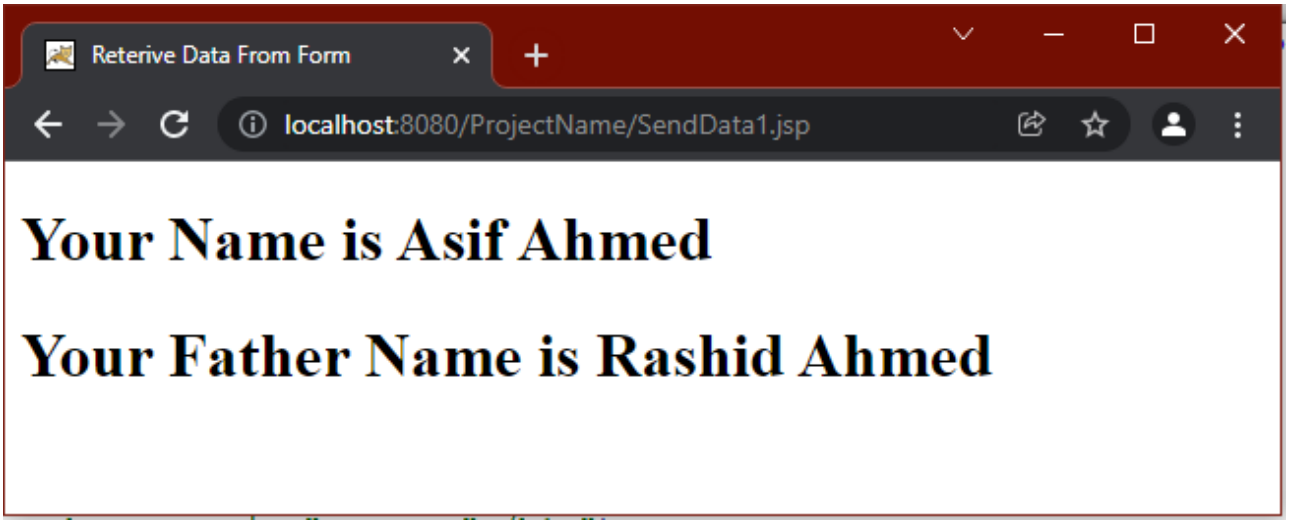
After the user file out field in the InpuForm1.html Web form and click the submit button using the post method, the your fields name (txtname, txtfname and submit) that were assigned to the controls in the "InputForm1.html" form automatically become in the request.getParameter("txtname") and request.getParameter("txtfname") , the values the user enters in the Your name and Father name input boxes ("Asif Ahmed" and "Rashid Ahmed"). in the following example and the value assigned to the submit button SendData1.jsp" become the values in the request.getParameters() method that can be accessed by the processing script.

Create a new document file in your text editor. Type the below script and save the document ***SendData1.jsp*** in the tomcat/webapps/ProjectName folder.

```
<%@ page contentType="text/html; charset=utf-8" language="java" import="java.sql.*" errorPage="" %>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Reterive Data From Form</title>
</head>
<%
 String Name,FName;
 Name = request.getParameter("txtname");
 FName=request.getParameter("txtfname");

 out.print("<h1> Your Name is "+Name+"</h1>");
 out.print("<h1> Your Father Name is "+FName+"</h1>");

%>
<body>
</body>
</html>
```

Then open the InputForm1.html file in the Web browser by entering the following URL http://localhost:8080/InputForm1.html and input the your name and father name and click the

Submit button then we will see output on the browser, bellow



Note: When form uses method ***get*** to transfer data, the data is visible in the URL Address bar as string, therefore the values are not hidden or secure.

If we use ***get*** method in form then, after click on submit button then result show on the browser and that come and show in the URL see below.

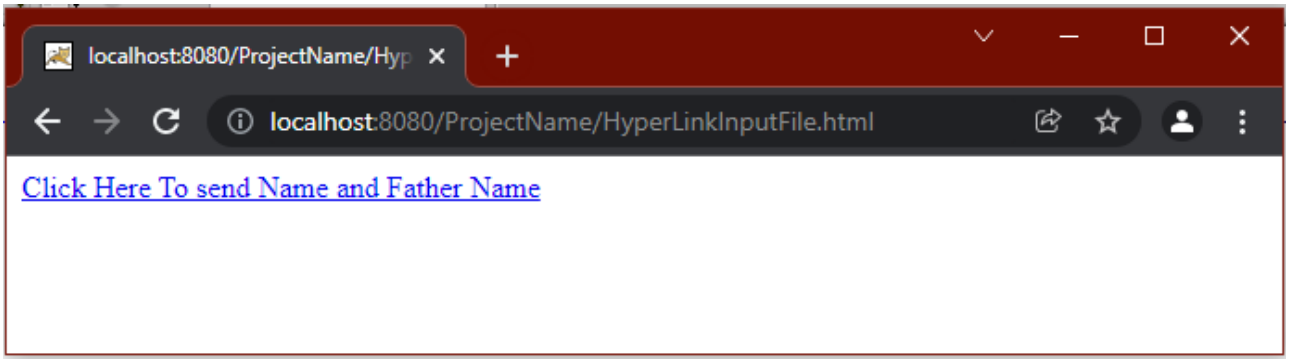
2. URL rewriting.

Send Data from Hyperlink with parameters:

Create a new document file in your text editor. Type the below script and save the document ***HyperLinkInputFile.html*** file in the tomcat/webapps/ProjectName folder. Assume we have an HTML page that contains a hyperlink with parameters:

```
<html>
<body>
<a href="RetrieveHyperLink.jsp?name=Asif&fname=Farhan">Click Here To send Name and Father Name
</a>
</body>
</html>
```

Then open the HyperLinkInputFile.html file in the Web browser by entering the following URL http://localhost:8080/HyperLinkInputFile.html now we will see following output on the browser screen.



Retrieving data from Hyperlink

Create a new document file in your text editor. Type the below script and save the document ***RetrieveHyperLink.jsp*** file in the ***tomcat/webapps/ProjectName*** folder.

```
<%@ page contentType="text/html; charset=utf-8" language="java" import="java.sql.*" errorPage="" %>
<!DOCTYPE html>
<html>
<head>
 <title> Rretrieve Data from Hyper link </title>
</head>
<body>
<%
 String Name,FName;
 Name = request.getParameter("name");
 FName=request.getParameter("fname");

 out.print("<h1> Your Name is "+Name+"</h1>");
 out.print("<h1> Your Father Name is "+FName+"</h1>");
%>
</body>
</html>
```

After Click then result show on the browser and that come and show in the URL see below

Note: **?** mark symbol is use to data send through by hyperlink variable and **&** symbol is use to more than one hyperlink variables.

3. Session

Session object represents a buffer on the server that can be used to maintain state between multiple requests made by client. For each user a session object is provided by the user. Session is represented by the implementation of javax.servlet.http.HttpSession interface.

Create Session Object

So let's take a look at some of the coding methods we can use to manipulate a session object. Here's how we add data to a session object:

Syntax

session.setAttribute(String name, Object value)

Example

session.setAttribute("UserId","Muhammad Ali");

Retrieve Session

Remember, you can place any kind of data into the session object, may it be integers or your custom objects. Next, we take a look at how to retrieve data from the session object:

Syntax

objectName = session.getAttribute(String name);

Example

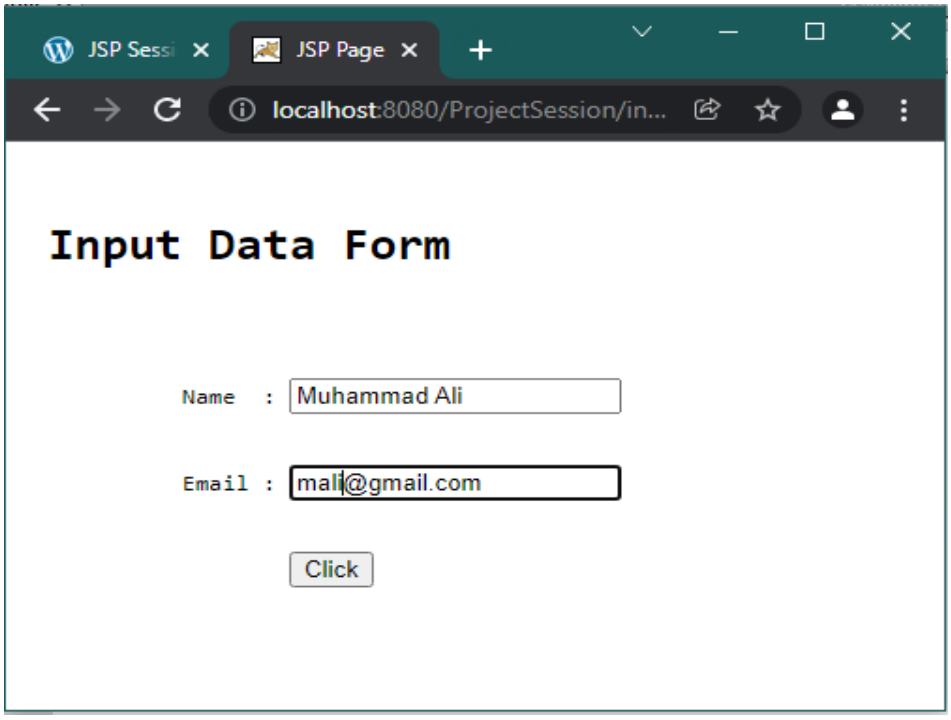
String UserName = session.getAttribute(UserId);

Some other JSP Session Methods	Description
isNew()	The return type of this method is Boolean and it returns true if the session is true.
getId()	The return type of this method is String and it returns the ID of the session object.
invalidate()	The return type of this method is void and it invalidates the session object, unbinding all data associated to it.
setMaxInactiveIntervals(long mills)	The return type of this method is void and it sets the idle time for a session object to expire. This method takes milliseconds as argument.
removeAttribute(String name)	This method removes the object bound with the specified name from this session.

Let's get understand it by an example. Create index.jsp file in your project like below

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <pre>
    <form action="CreateSessionFile.jsp" method="post">
      Name  : <input type="text" name="uname">

      Email : <input type="text" name="email">

          <input type="submit" value="Click"><br/>
    </pre>
  </form>
</body>
</html>
```

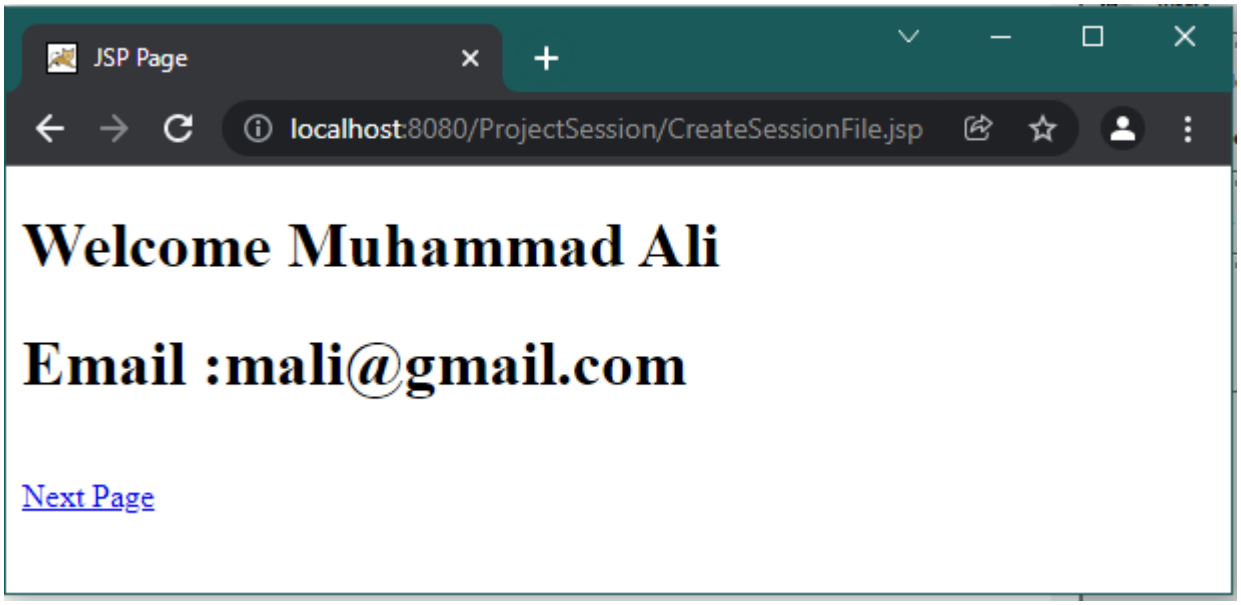
#Output: The output of this index.jsp file will look like below image.



#CreateSessionFile.jsp: From here we will set attributes in our session, check below code.

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <%

      String name = request.getParameter("uname");
      String email=request.getParameter("email");
      out.print(" <h1> Welcome  " + name+"</h1>")
      out.print(" <h1> Email :"+email+"</h1>"));


      session.setAttribute("user", name);     // create or set attribute in session object
      session.setAttribute("email", email);


    %>
    <br>
    <a href = "LinkOtherFile.jsp">Next Page</a>
  </body>
</html>
```

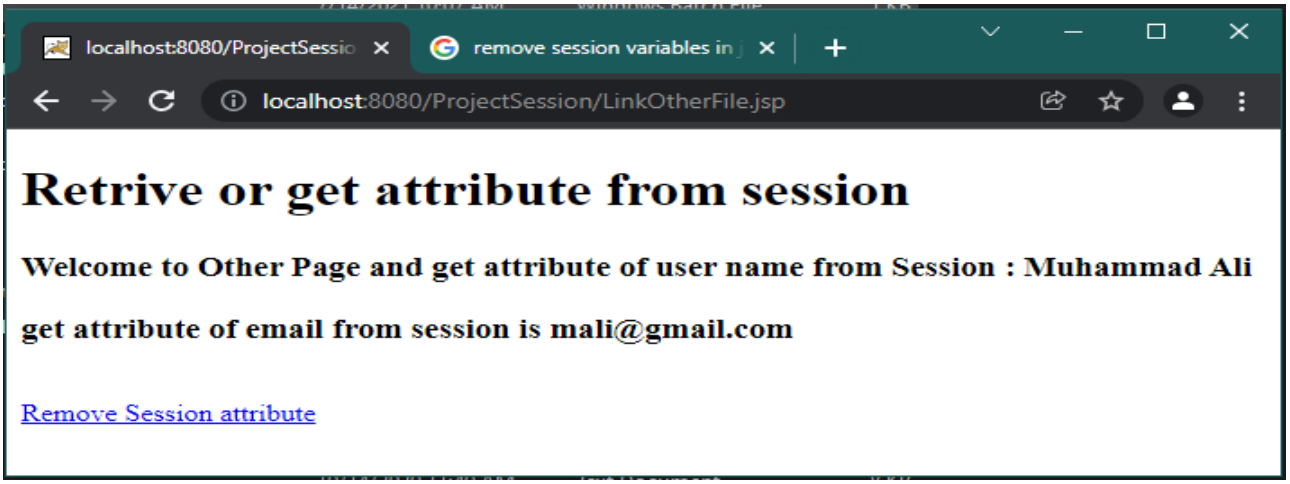
#Output: When you will click on index.jsp's "click" button so this code will run and the output of this above code will look like below image and also we have set attributes in session from above code.



Here click on Next Page then run LinkOtherFile.jsp so this code will run and the output of this above code will look like below image and also we have get or retrieve attributes in session from above code.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<body>
<h1> Retrive or get attribute from session </h1>
<%

String name=(String)session.getAttribute("user");    //Getting Session Attribute
String email=(String)session.getAttribute("email");
out.print("<h3>Welcome to Other Page and get attribute of user name from Session : "+name+"</h3>");
out.print("<h3>get attribute of email from session  is "+email+"</h3>");

%>
</body>
</html>
```

The output of this code will look like below image.



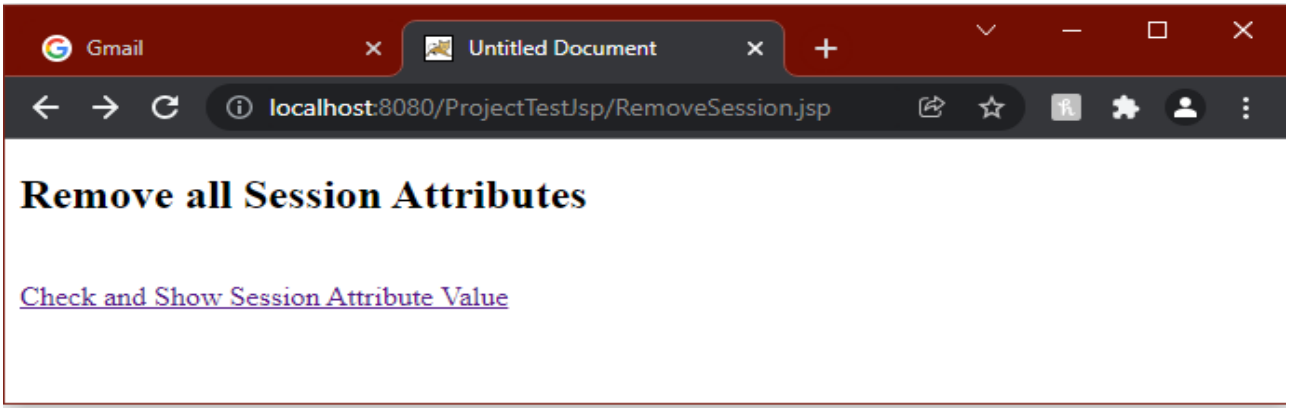
How to delete Session data in JSP?

1) Remove a particular attribute − You can call the public void ***removeAttribute(String name***) method to delete the value associated with the particular key.
2) Delete the whole session − You can call the public void **invalidate()** method to discard an entire session.

Here click on Remove Session Attribute hyperlink then run RemoveSession.jsp so this code will run and the output of this given below code will look like below image and all session attribute remove.

Example

```
<%@ page contentType="text/html; charset=utf-8" language="java" import="java.sql.*"
errorPage="" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
</head>
<%
 session.invalidate();

 out.println("<h2> Remove all Session Attributes </h2>");
%>
<br />
<a href="LinkOtherFile.jsp"> Check and Show Session Attribute Value </a>
<body>
</body>
</html>
```



4. Cookie

Cookies are text files stored on the client computer and they are kept for various information tracking purposes. JSP transparently supports HTTP cookies using underlying servlet technology. Browser stores this information on the local machine for future use.

There are three steps involved in identifying and returning users −

1) Server script sends a set of cookies to the browser. For example, name, age, or identification number, etc.
2) Browser stores this information on the local machine for future use.
3) When the next time the browser sends any request to the web server then it sends those cookies information to the server and server uses that information to identify the user or may be for some other purpose as well.

Setting Cookies with JSP

Setting cookies with JSP involves three steps −

Step 1: Creating a Cookie Object You call the Cookie constructor with a cookie name and a cookie value, both of which are strings.

Syntax

Cookie cookie = new Cookie("key","value");

Keep in mind, neither the name nor the value should contain white space or any of the following characters − [ ] ( ) = , " / ? @ : ;

Step 2: Setting the maximum age. You use setMaxAge(second) to specify how long (in seconds) the cookie should be valid. The following code will set up a cookie for 2 hours.

Syntax:

```
cookie.setMaxAge(60*60*2);
```

Step 3: Sending the Cookie into the HTTP response headers. You use response.addCookie() to add cookies in the HTTP response header as follows

Syntax

```
response.addCookie(cookie);
```

Example

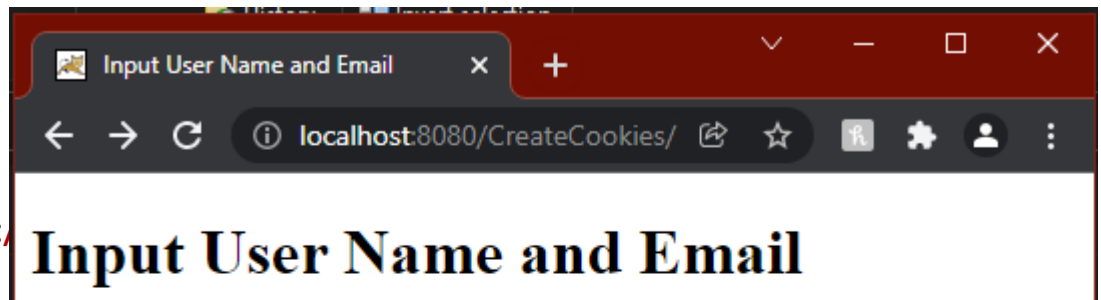
In this JSP cookies example, we will learn how to call cookie constructor in JSP by creating cookies of username and email, and add age to the cookie for 1 hours and trying to get the variable names in the following example.

Here we are taking a form that name ***index.html*** which has to be processed in ***CreateCookies.jsp*** Also, we are taking two fields **"username"** and **"email"** which has to be taken input from the user with a submit button.

Index.html

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Input Form</title>
</head>
<body>
<h1> Input User Name and Email </h1>
<form action="CreateCookies.jsp" method="GET">
<pre>
      Username: <input type="text" name="username"/>

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

            <input type="submit" value="Submit" />
</pre>
</form>
</body>
</html>
```

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

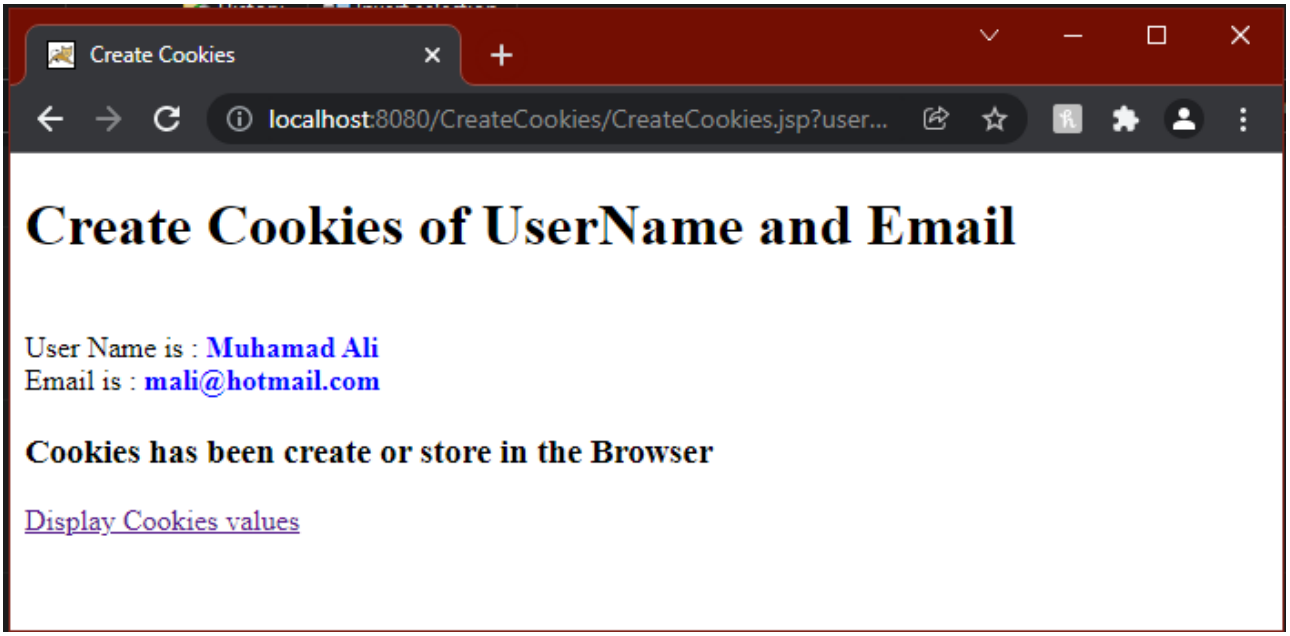


Here first get usernamevalue and emailvalue using by request.getParameter() method then creating two cookie objects of “usernameobj” and “emailobj” using request.getParameter()

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
  <title> Create Cookies </title>
<style>
 b{
            color:blue;
 }
</style>
</head>
<body>
<h1> Create Cookies of UserName and Email </h1>
<%
  String usernamevalue,emailvalue;

  usernamevalue=  request.getParameter("username");
  emailvalue = request.getParameter("email");

  Cookie usernameobj = new Cookie("username", usernamevalue );
  Cookie emailobj = new Cookie("email", emailvalue);

  // cookie for 1 minut interval
  usernameobj.setMaxAge(60);
  emailobj.setMaxAge(60);

  // Add both the cookies in the response header.
  response.addCookie( usernameobj );
  response.addCookie( emailobj );
```

```
  out.println("<br>User Name is  : <b>"+usernamevalue+"</b>");
  out.println("<br>Email is      : <b>"+emailvalue+"</b>");

%>
<h3> Cookies has been create or store in the Browser </h3>
<a href="DisplayCookies.jsp" > Display Cookies values  </a>
</body>
</html>
```

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

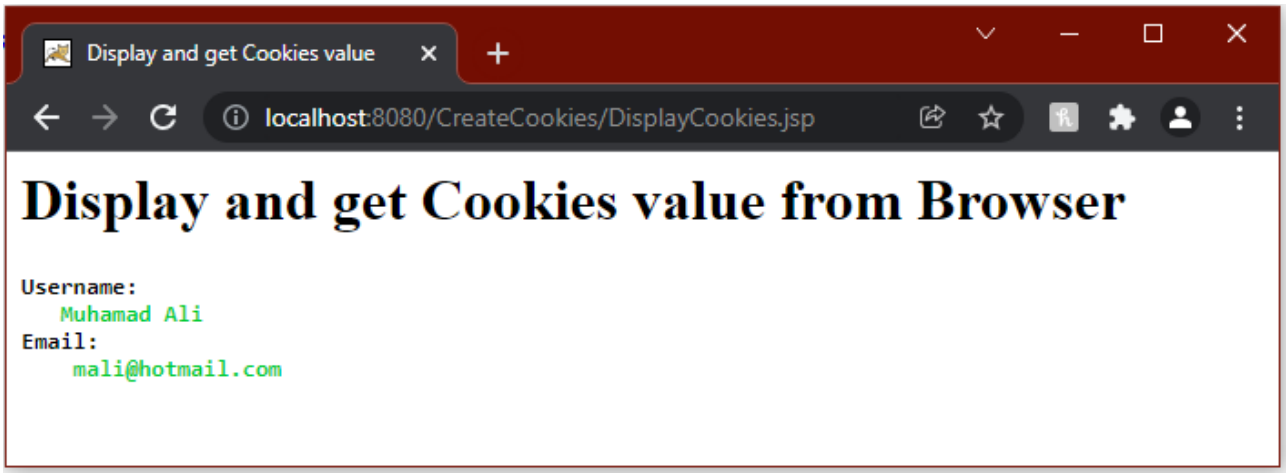


In this code here create cookie object of array that name is ***cookieobject***. Two cookies and can fetched when requested by ***reguest.getCookies().***

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Display and get Cookies value</title>
</head>
<style>
strong
{color:#0C3;
}


</style>
```

```
<body>
<h1>Display and get Cookies value from Browser </h1>
<%
Cookie cookieobject[]=request.getCookies();

        String name= cookieobject [0].getValue();


%>
<pre>
<b>Username:</b>
 <strong> <%= cookieobject [1].getValue() %> </strong>
<b>Email:</b>
  <strong> <%= cookieobject [2].getValue()%> </strong>

</pre>


</body>
</html>
```

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



Exercise

Theory Question

1) What is Managing State or HTTP Protocol?
2) What is Session state or object.
3) What is Cookie and write process step.
4) What is difference between the get and post method.

Practical Question

1) Write a web application to input your name, Roll number and technology and display all given input information on the next page.
2) Write a web application to input your Email, Full Name, and image and display all given formation on the next page by using session and cookie techniques.

Objective and MCQ

1) _____is a stateless (or non-persistent) protocol.
   a) IP
   b) FTP
   c) HTTP
   d) MTTP

2) When form uses method ___ to transfer data, the data is visible in the URL Address bar as string.
   a) get
   b) post
   c) delete
   d) top

3) ____ Symbol is use to data send through by hyperlink variable.
   a) %
   b) ?
   c) #
   d) @

4) _____ Symbol is use to more than one hyperlink variables.
   a) *
   b) ?
   c) #
   d) &

5) _____object represents a buffer on the server that can be used to maintain state between multiple requests made by client.
   a) Cookie
   b) Hyperlink
   c) URL rewriting
   d) Session

6) _____are text files stored on the client computer t.
   a) Cookies
   b) Hyperlink
   c) URL rewriting
   d) Session

7) Delete the whole session you can call the public _____ method to discard an entire session.
   a) invalidate()
   b) getId()
   c) isNew()
   d) getid()

8) Which you use _____ to specify how long (in seconds) the cookie should be valid.
   e) setMaxAge(second)
   f) getMaxAge(second)
   g) setMaxTime second ()
   h) getMaxInterval(second)