ICON OF JAVA

ADV. JAVA

Servlets

9. Session Tracking Mechanism

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

Session Tracking Mechanisms

As part of the web application development it is essential to manage the clients previous request data at the time of processing later request.

To achieve the above requirement if we use request object then container will create request object when it receives request from client and container will destroy request object when it dispatch response to client.

Due to this reason request object is not sufficient to manage clients previous request data at the time of processing later request.

To achieve the above requirement we able to use ServletContext object, but ServletContext object will share its data to all the components which are used in the present applications and to all the users of the present web application.

Due to this reason ServletContext object is unable to provide clear cut separation between multiple users.

In web applications, to manage clients previous request data at the time of processing later request and to provide separation between multiple users we need a set of mechanisms explicitly at server side called as **Session Tracking Mechanisms.**

Session:

Session is a time duration, it will start from the starting point of client conversation with server and will terminate at the ending point of client conversation with the server.

The data which we transferred from client to server through multiple number of requests during a particular session then that data is called **State of the Session.**

In general in web applications, container will prepare a request object similarly to represent a particular user we have to prepare a separate session.

If we allow multiple number of users on a single web application then we have to prepare multiple number of session objects.

In this context, to keep track of all the session objects at server machine we need a set of explicit mechanisms called as **Session Tracking Mechanisms.**

In web applications, there are 4 types of Session Tracking Mechanisms.

1. HttpSession Session Tracking Mechanism
2. Coockies Session Tracking Mechanism
3. URL-Rewriting Session Tracking Mechanism
4. Hidden Form Fields Session Tracking Mechanism

From the above Session Tracking Mechanisms Servlet API has provided the first 3 Session Tracking Mechanisms as official mechanisms, Hidden Form Fields Session Tracking Mechanism is purely developers creation.

1. HttpSession Session Tracking Mechanism:

In **HttpSession Session Tracking Mechanism**, we will create a separate HttpSession object for each and every user, at each and every request we will pick up the request parameters from request object and we will store them in the respective HttpSession object for the sake of future reusability.

After keeping the request parameters data in HttpSession object we have to generate the next form at client browser by forwarding the request to particular static page or by generating dynamic form.

In HttpSession Session Tracking Mechanism, to create HttpSession object we will use either of the following methods.

1. req.getSession();
2. req.getSession(false);

Q: What is the difference between getSession() method and getSession(false) method?

Ans: Both the methods can be used to return HttpSession object.

To get HttpSession object if we getSession() method then container will check whether any HttpSession object existed for the respective user or not, if any HttpSession object is existed then container will return the existed HttpSession object reference.

If no HttpSession object is existed for the respective user then container will create a new HttpSession object and return its reference.

public HttpSession getSession()

Ex: HttpSession hs=req.getSession();

To get HttpSession object if we getSession(false) method then container will check whether any HttpSession object existed w.r.t. user or not, if any HttpSession object is existed then container will return that HttpSession object reference.

If no HttpSession object is existed then container will return null value without creating new HttpSession object.

public HttpSession getSession(boolean b)

Ex: HttpSession hs=req.getSession(false);

Note: getSession(true) method functionality is almost all same as getSession() method.

Q: If we allow multiple number of users to access present web application then automatically container will create multiple number of HttpSession objects. In this case how container will identify user specific HttpSession object in order to put user specific attributes and to get attributes?

Ans: In HttpSession Session Tracking Mechanism, when we create HttpSession object automatically container will create an unique identification number in the form of hexadecimal number called as **Session Id.**Container will prepare session id in the form of Cookie with the name **JSESSIONID.**

In general the basic nature of Cookie is to transfer from server to client automatically along with response and it will be transferred from client to server automatically along with request.

Due to the above nature of Cookies session id Cookie will be transferred from server to client and from client to server automatically.

In the above context, if we use getSession() method or getSession(false) method first container will pick up session id value from request and it will identify the user specific HttpSession object on the basis of session id value.

To destroy HttpSession object explicitly we will use the following method from HttpSession.

public void invalidate()

If we want to destroy HttpSession object after a particular ideal time duration then we have to use the following method.

public void setMaxInactiveInterval(int time)

In web applications, HttpSession object will allow only attributes data, it will not allow parameters data.

To set an attribute on to the HttpSession object we have to use the following method.

public void setAttribute(String name, Object value)

To get a particular attribute value from HttpSession object we have to use the following method.

public Object getAttribute(String name)

To get all attribute names from HttpSession object we have to use the following method.

public Enumeration getAttributeNames()

To remove an attribute from HttpSession object we have to use the following method.

public void removeAttribute(String name)

Drawbacks:

1. In HttpSession Session Tracking Mechanism, we will create a separate HttpSession object

for each and every user, where if we increase number of users then automatically number of HttpSession object will be created at server machine, it will reduce the overall performance of the web application.

2. In case of HttpSession Session Tracking Mechanism, we are able to identify user specific HttpSession object among multiple number of HttpSession objects by carrying Session Id value from client to server and from server to client.

In the above context, if the client browser disable Cookies then HttpSession Session Tracking Mechanism will not execute its functionality.

httpsessionapp:-

form1.html:

```
<html>
<bodybgcolor="pink">
<center>
      <formmethod="get"action="first">
            Name : <inputtype="text"name="uname"/><br><br>
            Age : <inputtype="text"name="uage"/><br><br>
            <inputtype="submit"value="Next"/>
      </form>
</center>
</body>
</html>
```

form2.html:

```
<html>
<bodybgcolor="pink">
<center>
	<formmethod="get"action="second">
		Qualification : <inputtype="text"name="uqual"/><br><br>
		Designation : <inputtype="text"name="udesig"/><br><br>
		<inputtype="submit"value="Next"/>
	</form>
</center>
</body>
</html>
```

form3.html:

```
<html>
<bodybgcolor="pink">
<center>
	<formmethod="get"action="display">
		Email : <inputtype="text"name="email"/><br><br>
		Mobile : <inputtype="text"name="mobile"/><br><br>
		<inputtype="submit"value="Display"/>
	</form>
</center>
</body>
</html>
```

web.xml:

```
<web-app>
<display-name>httpsessionapp</display-name>
<welcome-file-list>
<welcome-file>form1.html</welcome-file>
</welcome-file-list>
<servlet>
<servlet-name>FirstServlet</servlet-name>
<servlet-class>FirstServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>FirstServlet</servlet-name>
<url-pattern>/first</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>SecondServlet</servlet-name>
<servlet-class>SecondServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>SecondServlet</servlet-name>
<url-pattern>/second</url-pattern>
</servlet-mapping>
```

```
<servlet>
<servlet-name>DisplayServlet</servlet-name>


<servlet-class>DisplayServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>DisplayServlet</servlet-name>
<url-pattern>/display</url-pattern>
</servlet-mapping>
</web-app>
```

FirstServlet.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;
import javax.servlet.http.HttpSession;

publicclass FirstServlet extends HttpServlet {

	protectedvoid doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		String uname=request.getParameter("uname");
		String uage=request.getParameter("uage");
		HttpSession hs=request.getSession();
		hs.setAttribute("uname", uname);
		hs.setAttribute("uage", uage);
		RequestDispatcher rd=request.getRequestDispatcher("form2.html");
		rd.forward(request, response);
	}
}
```

SecondServlet.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;
import javax.servlet.http.HttpSession;

publicclass SecondServlet extends HttpServlet {
```

```
	protectedvoid doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

		String uqual=request.getParameter("uqual");
		String udesig=request.getParameter("udesig");
		HttpSession hs=request.getSession();
		hs.setAttribute("uqual", uqual);
		hs.setAttribute("udesig", udesig);
		RequestDispatcher rd=request.getRequestDispatcher("form3.html");
		rd.forward(request, response);
	}
}
```

DisplayServlet.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;
import javax.servlet.http.HttpSession;

publicclass DisplayServlet extends HttpServlet {

	protectedvoid doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		response.setContentType("text/html");
		PrintWriter out=response.getWriter();
		String email=request.getParameter("email");
		String mobile=request.getParameter("mobile");
		HttpSession hs=request.getSession();
		String uname=(String)hs.getAttribute("uname");
		String uage=(String)hs.getAttribute("uage");
		String uqual=(String)hs.getAttribute("uqual");
		String udesig=(String)hs.getAttribute("udesig");
		out.println("<html>");
		out.println("<body bgcolor='lightgreen'>");
		out.println("<center><br><br>");
		out.println("<table bgcolor='lightyellow'>");
		out.println("<tr><td colspan='2'><center><b><font size='5'
color='red'><u>Registration Details</u></font></b></center></td></tr>");
		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>Qualification</td><td>"+uqual+"</td></tr>");
		out.println("<tr><td>Designation</td><td>"+udesig+"</td></tr>");
		out.println("<tr><td>Email</td><td>"+email+"</td></tr>");
		out.println("<tr><td>Mobile</td><td>"+mobile+"</td></tr>");
		out.println("<tr><td>Status</td><td>Success</td></tr>");
		out.println("</table></center>");
		out.println("</body>");
```

```
                out.println("</html>");
        }
}
```

2. Cookies Session Tracking Mechanism:

Cookie is a small object, it can be used to represent a single name value pair and which will be maintained permanently at client machine.

In HttpSession Session Tracking Mechanism, all the clients conversations will be maintained at server machine in the form of HttpSession objects.

In HttpSession Session Tracking Mechanism, if we increase number of users then automatically number of HttpSession objects will be created at

server. So that HttpSession Session Tracking Mechanism will increase burden to server machine.

To overcome the above problem we have to use an alternative mechanism, where we have to manage all the clients conversations at the respective client machines only.

To achieve the above requirement we have to use **Cookies Session Tracking Mechanism.**

In Cookies Session Tracking Mechanism, at each and every client we will pick up all the request parameters, prepare a separate Cookie for each and every request parameter, add all the Cookies to response object.

In the above context, when container dispatch response to client automatically all the added Cookies will be transferred to client and maintain at client machine permanently.

In the above context, when we send further request from the same client automatically all the Cookies will be transferred to server along with request.

By repeating the above process at each and every request we are able to manage clients previous data at the time of processing later request.

Drawbacks:

1. If we disable the Cookies at client browser then Cookies Session Tracking Mechanism will not execute its functionality.
2. In case of Cookies Session Tracking Mechanism, all the clients data will be maintain at the respective client machine only, which is open to every user of that machine. So that Cookies Session Tracking Mechanism will not provide security for the application data.

cookiesapp:-

form1.html:

```
<html>
<bodybgcolor="lightgreen">
<center>
<formaction="./first">
<br><br><br>
<h1>Product Registration Form</h1><br><br>
<h2>
        Product Id <inputtype="text"name="pid"/><br><br>
        Product Name <inputtype="text"name="pname"/><br><br>
        <inputtype="submit"value="Next"/>
</h2>
</form></center></body>
</html>
```

form2.html:

```
<html>
<bodybgcolor="lightgreen">
```

```
<center>
<formaction="./second">
<br><br><br>
<h1>Product Registration Form(Cont..)</h1><br><br>
<h2>
        Product Cost <inputtype="text"name="pcost"/><br><br>
        Product Quantity <inputtype="text"name="pquantity"/><br><br>
        <inputtype="submit"value="Next"/>
</h2>
</form></center></body>
</html>
```

form3.html:

```
<html>
<bodybgcolor="lightgreen"><center>
<formaction="./reg">
<br><br><br>
<h1>Product Registration Form(Cont..)</h1><br><br>
<h2>
        Manufactured Date <inputtype="text"name="man_Date"/><br><br>
        Expiry Date <inputtype="text"name="exp_Date"/><br><br>
        <inputtype="submit"value="Registration"/>
</h2>
</form></center></body>
</html>
```

web.xml:

```
<web-app>
<display-name>cookiesapp</display-name>
<welcome-file-list>
<welcome-file>form1.html</welcome-file>
</welcome-file-list>
<servlet>
<servlet-name>FirstServlet</servlet-name>
<servlet-class>FirstServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>FirstServlet</servlet-name>
<url-pattern>/first</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>SecondServlet</servlet-name>
<servlet-class>SecondServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>SecondServlet</servlet-name>
```

```
<url-pattern>/second</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>RegistrationServlet</servlet-name>
<servlet-class>RegistrationServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>RegistrationServlet</servlet-name>
<url-pattern>/reg</url-pattern>
</servlet-mapping>
</web-app>
```

FirstServlet.java:

```
import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class FirstServlet extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                String pid=request.getParameter("pid");
                String pname=request.getParameter("pname");


                Cookie c1=new Cookie("pid", pid);
                Cookie c2=new Cookie("pname", pname);
                response.addCookie(c1);
                response.addCookie(c2);
                RequestDispatcher rd=request.getRequestDispatcher("form2.html");
                rd.forward(request, response);
        }
    }
```

SecondServlet.java:

```
import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
public class SecondServlet extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		try{
			int pcost=Integer.parseInt(request.getParameter("pcost"));
			int
pquantity=Integer.parseInt(request.getParameter("pquantity"));
			Cookie c3=new Cookie("pcost", ""+pcost);
			Cookie c4=new Cookie("pquantity", ""+pquantity);
			response.addCookie(c3);
			response.addCookie(c4);
			RequestDispatcher
rd=request.getRequestDispatcher("form3.html");
			rd.forward(request, response);
		}
		catch (Exception e) {
			e.printStackTrace();
		}
	}
}
```

RegistrationServlet.java:

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

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


import javax.servlet.http.HttpServletResponse;

public class RegistrationServlet extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		response.setContentType("text/html");
		PrintWriter out=response.getWriter();
		String man_Date=request.getParameter("man_Date");
		String exp_Date=request.getParameter("exp_Date");
		Cookie[] c=request.getCookies();
		String pid=c[0].getValue();
		String pname=c[1].getValue();
		int pcost=Integer.parseInt(c[2].getValue());
		int pquantity=Integer.parseInt(c[3].getValue());
		ProductBean pb=new ProductBean();
		String
status=pb.register(pid,pname,pcost,pquantity,man_Date,exp_Date);
```

```
                out.println("<html>");
                out.println("<body bgcolor='pink'>");
                out.println("<center><br><br>");
                out.println("<u>Product Registration Details</u><br><br>");
                out.println("Product Id....."+pid+"<br><br>");
                out.println("Product Name....."+pname+"<br><br>");
                out.println("Product Cost....."+pcost+"<br><br>");
                out.println("Product Quantity....."+pquantity+"<br><br>");
                out.println("Product Manufactured
        Date....."+man_Date+"<br><br>");
                out.println("Product Expiry Date....."+exp_Date+"<br><br>");
                out.println("Status....."+status);
                out.println("</center>");
                out.println("</body>");
                out.println("</html>");
            }
}
```

ProductBean.java:

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

        public class ProductBean {
            Connection con;
            Statement st;
            ResultSet rs;
            String status="";
            public ProductBean(){
                try{
                    Class.forName("oracle.jdbc.driver.OracleDriver");


            con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","sy
        stem","durga");
                    st=con.createStatement();
                }
                catch (Exception e) {
                    e.printStackTrace();
                }
            }
            public String register(String pid,String pname,int pcost,int pquantity,String
        man_Date,String exp_Date){
                try{
                    rs=st.executeQuery("select * from product where
        pid='"+pid+"'");
                    boolean b=rs.next();
                    if(b==true){
                        status="Product Existed Already";
                    }
```

```
                        else{
                                int rowCount=st.executeUpdate("insert into product
        values('"+pid+"','"+pname+"',"+pcost+","+pquantity+",'"+man_Date+"','"+exp_Da
        te+"')");
                                if(rowCount==1){
                                        status="SUCCESS";
                                }
                                else{
                                        status="FAILURE";
                                }
                        }
                }
                catch (Exception e) {
                        status="FAILURE";
                        e.printStackTrace();
                }
                return status;
        }
}
```


To represent a Cookie in web application Servlet API has provided a predefined class in the form of javax.servlet.http.Cookie.

To create Cookie object with a particular name-value pair we have to use the following Cookie class constructor.

public Cookie(String name, String value)

To add a Cookie to the response object we have to use the following method from HttpServletResponse.

public void addCookie(Cookie c)

To get all the Cookies from response object we need to use the following method.

public Cookie[] getCookies()

To get name and value of a Cookie we have to use the following methods,

public String getName()

public String getValue()

In web applications, it is possible to provide comments to the Cookies. So that to set the comment to Cookie and get the comment from Cookie we need to use the following methods.

public void setComment(String comment)

public String getComment()

In web applications, it is possible to provide version numbers to the Cookies. So that to set a version number to Cookie and get a version number from Cookie we need to use the following methods.

public void setVersion(int version_no)

public int getVersion()

In web applications, it is possible to specify life time to the Cookies. So that to set max age to Cookie and get max age from Cookie we need to use the following methods.

public void setMaxAge(int age)

public int getMaxAge()

In web applications, it is possible to provide domain names to the Cookies. So that to set domain name to Cookie and get domain name from Cookie we need to use the following methods.

public void setDomain(String domain)

public String getDomain()

In web applications, it is possible to provide a particular path to the Cookies to store. So that to set a path to Cookie and get a path from Cookie we need to use the following methods.

public void setPath(String path)

public String getPath()

3. URL-Rewriting Session Tracking Mechanism:

In case of HttpSession Session Tracking Mechanism, when we create HttpSession object automatically Session Id will be created in the form of the Cookie, where Session Id Cookie will be transferred from server to client and from client to server along with response and request automatically.

In HttpSession Session Tracking Mechanism, we are able to identify user specific HttpSession object on the basis of Session Id only.

In this context, if we disable Cookies at client browser then HttpSession Session Tracking Mechanism will not execute its functionality.

In case of Cookies Session Tracking Mechanism, the complete client conversation will be stored at the respective client machine only in the form of Cookies, here the Cookies data will be opened to every user of that machine. So that Cookies Session Tracking Mechanism will not provide security for the application data.

To overcome the above problem, we have to use URL-Rewriting Session Tracking Mechanism.

In case of URL-Rewriting Session Tracking Mechanism, we will not maintain the clients conversation at the respective client machine, we will maintain the clients conversationin the form of HttpSession object at server machine. So that URL-Rewriting Session Tracking Mechanism is able to provide very good security for the application data.

URL-Rewriting Session Tracking Mechanism is almost all same as HttpSession Session Tracking Mechanism, in URL-Rewriting Session Tracking Mechanism we will not depending on a Cookie to maintain Session Id value, we will manage Session Id value as an appender to URL in the next generated form.

In this context, if we send a request from the next generated form automatically the appended Session Id value will be transferred to server along with the request.

In this case, eventhough if we disable Cookies at client browser, but still we are able to get Session Id value at server machine and we are able to manage clients previous request data at the time of processing the later request.

In URL-Rewriting Session Tracking Mechanism, every time we need to rewrite the URL with Session Id value in the next generated form. So that this mechanism is called as **URL-Rewriting Session Tracking Mechanism.**

In URL-Rewriting Session Tracking Mechanism, it is mandatory to append Session Id value to the URL by getting Session Id value explicitly.

To perform this work HttpServletResponse has provided a separate method like,

```
public String encodeURL(String url)
```

Ex: out.println("<form method='get'

action='"+res.encodeURL("./second")+"'>");

Drawback:

In URL-Rewriting Session Tracking Mechanism, every time we need to rewrite the URL with Session Id value in the generated form, for this we must execute encodeURL() method. So that URL-Rewriting Session Tracking Mechanism should require dynamically generated forms, it will not execute its functionality with static forms.

urlrewritingapp:-

form1.html:

```
<html>
<bodybgcolor="lightgreen">
        <br><br><center><h1>Deposit Form</h1></center>
        <br><br><hr><br><br>
        <formaction="./first">
                <pre>
                        Account Number <inputtype="text"name="accNo"/>

                        Account Name   <inputtype="text"name="accName">

                        <inputtype="submit"value="Next">
                </pre>
        </form></body>
</html>
```

web.xml:

```
<web-app>
<display-name>urlrewritingapp</display-name>
<welcome-file-list>
<welcome-file>form1.html</welcome-file>
</welcome-file-list>
```

```
<servlet>
<servlet-name>FirstServlet</servlet-name>
<servlet-class>FirstServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>FirstServlet</servlet-name>
<url-pattern>/first</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>SecondServlet</servlet-name>
<servlet-class>SecondServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>SecondServlet</servlet-name>
<url-pattern>/second</url-pattern>
</servlet-mapping>

<servlet>
<servlet-name>DepositServlet</servlet-name>
<servlet-class>DepositServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>DepositServlet</servlet-name>
<url-pattern>/deposit</url-pattern>
</servlet-mapping>
</web-app>
```

FirstServlet.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;
import javax.servlet.http.HttpSession;

public class FirstServlet extends HttpServlet {

      protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
            PrintWriter out=response.getWriter();
            String accNo=request.getParameter("accNo");
            String accName=request.getParameter("accName");
            HttpSession session=request.getSession();
            session.setAttribute("accNo", accNo);
            session.setAttribute("accName", accName);
            out.println("<html><body bgcolor='cyan'><br><br>");
            out.println("<center><h1>Deposit
Form(Cont..)</h1></center><br><br><hr><br><br>");
            out.println("<form method='get'
```

```
action='"+response.encodeUrl("./second")+"'>");
                out.println("<pre>");
                out.println("Account Type    <input type='text' name='accType'/>");
                out.println();
                out.println("Account Branch <input type='text' name='accBranch'/>");
                out.println();
                out.println("  <input type='submit' value='Next'/>");
                out.println("</pre></form></body></html>");
        }
}
```

SecondServlet.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;
import javax.servlet.http.HttpSession;

public class SecondServlet extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                PrintWriter out=response.getWriter();
                String accType=request.getParameter("accType");
                String accBranch=request.getParameter("accBranch");
                HttpSession session=request.getSession();
                session.setAttribute("accType", accType);
                session.setAttribute("accBranch", accBranch);
                out.println("<html><body bgcolor='cyan'><br><br>");
                out.println("<center><h1>Deposit
Form(Cont..)</h1></center><br><br><hr><br><br>");
                out.println("<form method='get'
action='"+response.encodeUrl("./deposit")+"'>");
                out.println("<pre>");
                out.println("Depositor Name<input type='text' name='depName'/>");
                out.println();
                out.println("Deposit Amount <input type='text' name='depAmount'/>");
                out.println();
                out.println("  <input type='submit' value='Deposit'/>");
                out.println("</pre></form></body></html>");

        }
}
```

DepositServlet.java:

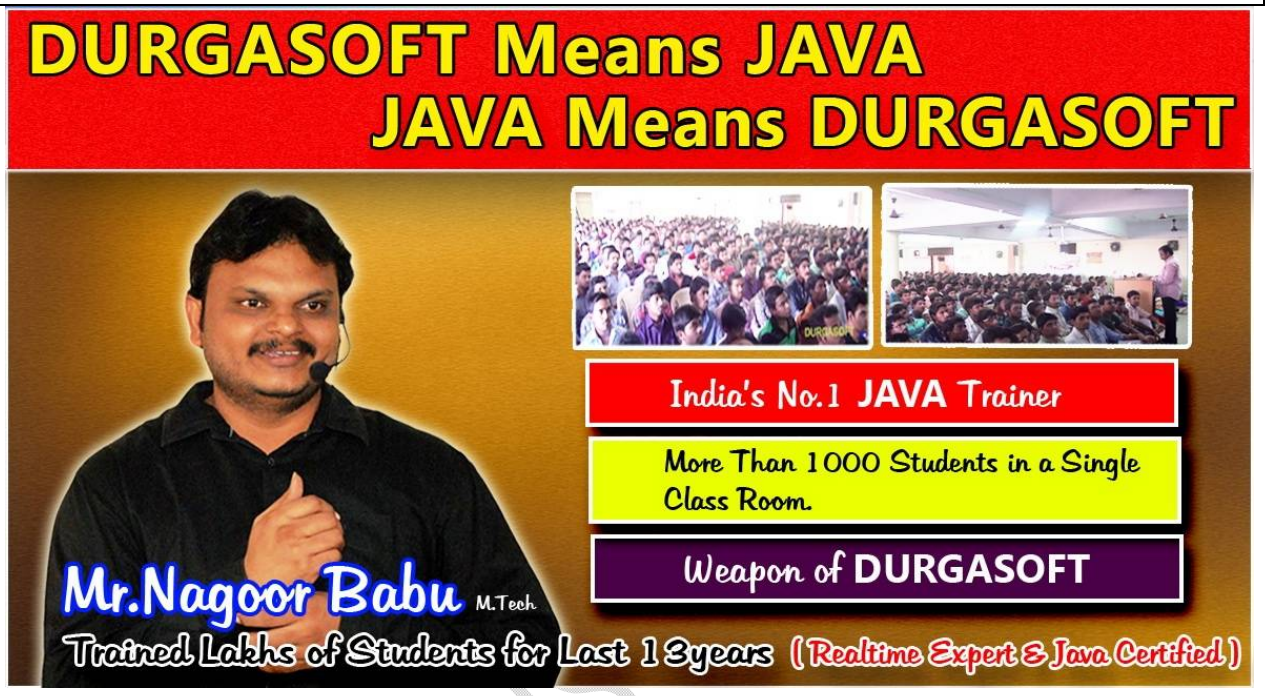
```
import java.io.IOException;
import java.io.PrintWriter;
```

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

public class DepositServlet extends HttpServlet {


	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		response.setContentType("text/html");
		PrintWriter out=response.getWriter();
		String depName=request.getParameter("depName");
		int depAmount=Integer.parseInt(request.getParameter("depAmount"));
		HttpSession session=request.getSession();
		String accNo=(String)session.getAttribute("accNo");
		String accName=(String)session.getAttribute("accName");
		String accType=(String)session.getAttribute("accType");
		String accBranch=(String)session.getAttribute("accBranch");
		TransactionBean tb=new TransactionBean();
		String status=tb.deposit(accNo,depAmount);
		out.println("<html><body bgcolor='lightblue'><br><br>");
		out.println("<center><table>");
		out.println("<tr><td colospan='2'><center><b><font size='6'
color='red'>");
		out.println("<u>Transaction
Details</u><font></b></center></td></tr>");
		out.println("<tr><td>Transaction Id</td><td>:t1</td></tr>");
		out.println("<tr><td>Transaction Name</td><td>:Deposit</td></tr>");
		out.println("<tr><td>Account Number</td><td>:"+accNo+"</td></tr>");
		out.println("<tr><td>Account Name</td><td>:"+accName+"</td></tr>");
		out.println("<tr><td>Account Type</td><td>:"+accType+"</td></tr>");
		out.println("<tr><td>Account
Branch</td><td>:"+accBranch+"</td></tr>");
		out.println("<tr><td>Depositor
Name</td><td>:"+depName+"</td></tr>");
		out.println("<tr><td>Deposit
Amount</td><td>:"+depAmount+"</td></tr>");
		int totalAvailableAmount=tb.getTotalAvailableAmount();
		out.println("<tr><td>Total Available
Amount</td><td>:"+totalAvailableAmount+"</td></tr>");
		out.println("<tr><td>Transaction
Status</td><td>:"+status+"</td></tr>");
		out.println("<tr><td colspan='2'><b><font size='5'>...Visit
Again...</td></tr>");
		out.println("</table></center></body></html>");
	}
}
```

TransactionBean.java:

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

public class TransactionBean {

	Connection connection;


	Statement statement;
	ResultSet resultSet;
	String status="";
	int totalAvailableAmount;

	TransactionBean(){
		try{
			Class.forName("oracle.jdbc.driver.OracleDriver");

	connection=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe",
"system", "durga");
			statement=connection.createStatement();
		}
		catch (Exception e) {
			e.printStackTrace();
		}
	}
	public String deposit(String accNo,int depAmount){
		try{
			int rowCount=statement.executeUpdate("update acc_details set
balance=balance+"+depAmount+"where accNo='"+accNo+"'");
			if(rowCount==1)
				status="SUCCESS";
			else
				status="FAILURE";
		}
		catch (Exception e) {
			status="FAILURE";
			e.printStackTrace();
		}
		return status;
	}
	public int getTotalAvailableAmount() {
		try{
			resultSet=statement.executeQuery("select * from acc_details");
			resultSet.next();
			totalAvailableAmount=resultSet.getInt(4);
		}
		catch (Exception e) {
```

```
                    e.printStackTrace();
                }
                return totalAvailableAmount;
        }
}
```



4. Hidden Form Field Session Tracking Mechanism:

Hidden Form Field Session Tracking Mechanism is not official Session Tracking Mechanism from Servlet API, it was purely developers creation.

In Hidden Form Field Session Tracking Mechanism, at each and every request we will pick up all the request parameters, generate dynamic form, in dynamic form generation we have to maintain the present request parameters data in the form of hidden fields.

In the above context, if we dispatch the response to client then we are able to get a dynamic form with visible fields and with invisible fields.

If we send a request from dynamic form then automatically all the visible fields data and invisible fields data will be send to server as request parameters.

By repeating above process at each and every request we are able to manage the clients previous request data at the time of processing the later request.

hiddenformfieldsapp:-

studentform.html:-

```
<html>
<head>
<bodybgcolor="lightgreen">
        <formmethod="get"action="./first">
        <center><b><br><br>
        Student Name:<inputtype="text"name="sname"/><br><br>
        Student Id:<inputtype="text"name="sid"/><br><br>
        Student Address:<inputtype="text"name="saddr"/><br><br>
        <inputtype="submit"value="Submit">
        </b></center>
        </form>
</body>
</html>
```

web.xml:-

```
<web-app>
<display-name>hiddenformfieldsapp</display-name>
<welcome-file-list>
<welcome-file>studentform.html</welcome-file>
</welcome-file-list>
<servlet>
<servlet-name>FirstServlet</servlet-name>
<servlet-class>FirstServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>FirstServlet</servlet-name>
<url-pattern>/first</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>SecondServlet</servlet-name>
<servlet-class>SecondServlet</servlet-class>
</servlet>


<servlet-mapping>
<servlet-name>SecondServlet</servlet-name>
<url-pattern>/second</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>ThirdServlet</servlet-name>
<servlet-class>ThirdServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>ThirdServlet</servlet-name>
<url-pattern>/third</url-pattern>
</servlet-mapping>
</web-app>
```

FirstServlet.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 FirstServlet extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		PrintWriter out=response.getWriter();
		String sname=request.getParameter("sname");
		String sid=request.getParameter("sid");
		String saddr=request.getParameter("saddr");
		out.println("<html><body bgcolor='lightyellow'>");
		out.println("<center><b><br><br>");
		out.println("Welcome to Student Application");
		out.println("<br><br>");
		out.println("<form method='get' action='/hiddenformfieldsapp/second'>");
		out.println("<input type='hidden' name=sname value='"+sname+"'>");
		out.println("<input type='hidden' name=sid value='"+sid+"'>");
		out.println("<input type='hidden' name=saddr value='"+saddr+"'>");
		out.println("<br><br>");
		out.println("Student Age:");
		out.println("<input type='text' name='sage'>");
		out.println("<br><br>");
		out.println("<input type='submit' value='Submit'>");
		out.println("</form></b></center></body></html>");

	}

}
```

SecondServlet.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 SecondServlet extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
```

```
                PrintWriter out=response.getWriter();
                String sname=request.getParameter("sname");
                String sid=request.getParameter("sid");
                String saddr=request.getParameter("saddr");
                String sage=request.getParameter("sage");
                out.println("<html><body bgcolor='lightyellow'>");
                out.println("<center><b><br><br>");
                out.println("Student Details Are...");
                out.println("<br><br>");
                out.println("Student Name....."+sname+"<br><br>");
                out.println("Student Id....."+sid+"<br><br>");
                out.println("Student Address....."+saddr+"<br><br>");
                out.println("<a href='/hiddenformfieldsapp/third?sage="+sage+"'>
                                                        SHOW STUDENT AGE</a>");
}
}
```

ThirdServlet.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 ThirdServlet extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                PrintWriter out=response.getWriter();
                String sage=request.getParameter("sage");
                out.println("<html><body bgcolor='lightpink'>");
                out.println("<center><b><br><br>");
                out.println("Student Age is....."+sage);
                out.println("</b></center></body></html>");
        }
}
```