Top of Form

#### Servlets Request & Response Interface

#### ServletRequest Interface

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.

#### ServletRequest Interface Methods

**Following are the important methods of ServletRequest Interface:**

Methods Description

public Object getAttribute(String name) Returns the value of the named attribute as an Object.

public int getContentLength( ) Returns the length of the request body in bytes.

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.

#### Example : Request to take username & password. Display username.

**//index.html**

<form action="test" method="post">

     User Name: <input type="text" name="uname"><br>

     Password: <input type = "password name = "password"><br>

     <input type="submit" value="Log In">

</form>

**//web.xml**

<servlet>

    <servlet-name>test</servlet-name>

    <servlet-class>ServletDemo</servlet-class>

</servlet>

<servlet-mapping>

    <servlet-name>test</servlet-name>

    <url-pattern>/test</url-pattern>

</servlet-mapping>

**//ServletDemo.java**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class ServletDemo extends HttpServlet

{

      protected void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException

      {

          response.setContentType("text/html");

          PrintWriter pw = response.getWriter();

          try

          {

              String username = req.getParameter("username");

              String password = req.getParameter("password");

              pw.println("<h1> Hello"+username+"<h1>");

          }

          finally

          {

              pw.close();

          }

      }

}

#### ServletResponse Interface

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.

#### ServletResponse Interface Methods

Methods Description

public void flushBuffer( ) It forces the content in the buffer to be written to the client.

public int getBufferSize( ) Returns the actual buffer size.

public String getContentType( ) Returns the content type used to send this response.

public PrintWriter getWriter( ) Returns the PrintWriter object that can be used to send character text to the client.

public void reset( ) Clears the buffer as well as status code and header data.

public void setBufferSize(int size) Sets the specified buffer size for the body of the response.

public void setContentType(String type) Sets the content type of the response being sent to the client.

Bottom of Form

Top of Form

#### Example of Registration form in servlet

Here, you will learn that how to create simple registration form in servlet. We are using oracle10g database. So you need to create a table first as given below:

1. CREATE TABLE  "REGISTERUSER"
2. (    "NAME" VARCHAR2(4000),
3. "PASS" VARCHAR2(4000),
4. "EMAIL" VARCHAR2(4000),
5. "COUNTRY" VARCHAR2(4000)
6. )
7. /

To create the registration page in servlet, we can separate the database logic from the servlet. But here, we are mixing the database logic in the servlet only for simplicity of the program. We will develop this page in JSP following DAO, DTO and Singleton design pattern later.

#### Example of Registration form in servlet

In this example, we have created the three pages.

* register.html
* Register.java
* web.xml

**register.html**In this page, we have getting input from the user using text fields and combobox. The information entered by the user is forwarded to Register servlet, which is responsible to store the data into the database.

1. <html>
2. <body>
3. <form action="servlet/Register" method="post">
5. Name:<input type="text" name="userName"/><br/><br/>
6. Password:<input type="password" name="userPass"/><br/><br/>
7. Email Id:<input type="text" name="userEmail"/><br/><br/>
8. Country:
9. <select name="userCountry">
10. <option>India</option>
11. <option>Pakistan</option>
12. <option>other</option>
13. </select>
15. <br/><br/>
16. <input type="submit" value="register"/>
18. </form>
19. </body>
20. </html>

**Register.java**This servlet class receives all the data entered by user and stores it into the database. Here, we are performing the database logic. But you may separate it, which will be better for the web application.

1. **import** java.io.\*;
2. **import** java.sql.\*;
3. **import** javax.servlet.ServletException;
4. **import** javax.servlet.http.\*;
6. **public** **class** Register **extends** HttpServlet {
7. **public** **void** doPost(HttpServletRequest request, HttpServletResponse response)
8. **throws** ServletException, IOException {
10. response.setContentType("text/html");
11. PrintWriter out = response.getWriter();
13. String n=request.getParameter("userName");
14. String p=request.getParameter("userPass");
15. String e=request.getParameter("userEmail");
16. String c=request.getParameter("userCountry");
18. **try**{
19. Class.forName("oracle.jdbc.driver.OracleDriver");
20. Connection con=DriverManager.getConnection(
21. "jdbc:oracle:thin:@localhost:1521:xe","system","oracle");
23. PreparedStatement ps=con.prepareStatement(
24. "insert into registeruser values(?,?,?,?)");
26. ps.setString(1,n);
27. ps.setString(2,p);
28. ps.setString(3,e);
29. ps.setString(4,c);
31. **int** i=ps.executeUpdate();
32. **if**(i>0)
33. out.print("You are successfully registered...");

36. }**catch** (Exception e2) {System.out.println(e2);}
38. out.close();
39. }
41. }

**web.xml file**The is the configuration file, providing information about the servlet.

1. <web-app>
3. <servlet>
4. <servlet-name>Register</servlet-name>
5. <servlet-**class**>Register</servlet-**class**>
6. </servlet>
8. <servlet-mapping>
9. <servlet-name>Register</servlet-name>
10. <url-pattern>/servlet/Register</url-pattern>
11. </servlet-mapping>
13. <welcome-file-list>
14. <welcome-file>register.html</welcome-file>
15. </welcome-file-list>
17. </web-app>

[download this example](https://static.javatpoint.com/src/servlet/registeruser1.zip)

[download the war file](https://static.javatpoint.com/src/servlet/registeruser.war)

[download this example (developed using Myeclipse IDE)](https://static.javatpoint.com/src/servlet/mregisteruser.zip)

[download this example (developed using Eclipse IDE)](https://static.javatpoint.com/src/servlet/eclipse/registeruser.zip)

[download this example (developed using Netbeans IDE)](https://static.javatpoint.com/src/servlet/netbeans/registeruser.zip)

To connect java application with the Oracle database ojdbc14.jar file is required to be loaded. Put this jar file in WEB-INF/lib folder.

[download the jar file ojdbc14.jar](https://static.javatpoint.com/src/jdbc/ojdbc14.jar)

**FOR OTHER EXAMPLE REFER TO THE SCROM PPT.**

Bottom of Form

Top of Form

**Video Link:**

**Reference Link:**

https://www.javatpoint.com/example-of-registration-form-in-servlet

https://www.javatpoint.com/example-of-fetching-result-for-the-given-rollno-in-servlet

https://www.javatpoint.com/improving-servlet-performance-to-fetch-records

https://www.geeksforgeeks.org/java-servlet-and-jdbc-example-insert-data-in-mysql/

https://www.tutorialspoint.com/servlets/servlets-database-access.htm

Bottom of Form