**SA - EXPERIMENT NO. 4**

**Java RMI Display Message APP**

1. **Create a Java project.**
2. **Create a package in the project and name it pkg\_RMI.**
3. **Create an interface and name it RMI\_interface. Add following code:**

**package** pkg\_RMI;

**import** java.rmi.Remote;

**import** java.rmi.RemoteException;

**public** **interface** RMI\_interface **extends** Remote{

**public** **void** displayMessage() **throws** RemoteException;

}

1. **Create a class and name it RMI\_Server. Add following code:**

**package** pkg\_RMI;

**import** java.rmi.server.UnicastRemoteObject;

**import** java.rmi.AlreadyBoundException;

**import** java.rmi.RemoteException;

**import** java.rmi.registry.LocateRegistry;

**import** java.rmi.registry.Registry;

**import** java.rmi.server.UnicastRemoteObject;

**public** **class** RMI\_Server **extends** UnicastRemoteObject **implements** RMI\_interface{

**public** RMI\_Server() **throws** RemoteException {

**super**();

}

**public** **static** **void** main(String[] args)**throws** RemoteException, AlreadyBoundException {

**try** {

Registry registry = LocateRegistry.*createRegistry*(1878);

registry.bind("hello", **new** RMI\_Server());

System.***out***.println("The RMI\_Server is running and ready...");

}

**catch** (Exception e) {

System.***out***.println("The RMI\_Server is not running...");

}

}

@Override

**public** **void** displayMessage()**throws** RemoteException{

System.***out***.println("----------------------------------------------");

System.***out***.println("Hello Drashti!");

System.***out***.println("----------------------------------------------");

}

}

1. **Create a class and name it RMI\_Client. Add following code:**

**package** pkg\_RMI;

**import** java.net.MalformedURLException;

**import** java.rmi.RemoteException;

**import** java.rmi.NotBoundException;

**import** java.rmi.Naming;

**public** **class** RMI\_Client {

**public** **static** **void** main(String[] args) **throws** MalformedURLException, RemoteException, NotBoundException{

**try** {

RMI\_interface helloAPI = (RMI\_interface) Naming.*lookup*("rmi://localhost:1878/hello");

helloAPI.displayMessage();

}

**catch**(Exception e)

{

System.***out***.println("The RMI APP is not running...");

e.printStackTrace();

}

}

}

1. **Run the server first.**
2. **Now run the client.**

**SA - EXPERIMENT NO. 3**

**Calculator Web service**

**Calculator.java – make changes in the doGet method**

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

// **TODO** Auto-generated method stub

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

out.println("<html><head><title>Servlet CalculatorServlet</title></head><body>");

double n1 = Double.*parseDouble*(request.getParameter("txtN1"));

double n2 = Double.*parseDouble*(request.getParameter("txtN2"));

double result = 0;

String opr = request.getParameter("opr");

if (opr.equals("+"))

result = n1 + n2;

if (opr.equals("\*"))

result = n1 \* n2;

if (opr.equals("-"))

result = n1 - n2;

if (opr.equals("/"))

result = n1 / n2;

out.println("<h1> Result = " + result);

out.println("</body></html>");

}

**Index.html**

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

</head>

<body>

<form action="Calculator">

Enter First Number <input type="text" name="txtN1"><br>

Enter Second Number <input type="text" name="txtN2" ><br>

Select an Operation<input type="radio" name="opr" value="+">

ADDTION <input type="radio" name="opr" value="-">

SUBSTRACTION <input type="radio" name="opr" value="\*">

MULTIPLY <input type="radio" name="opr" value="/">

DIVIDE <br><input type="reset">

<input type="submit" value="Calculate" >

</form>

</body>

</html>

**PalindromeCheckerServlet.java – make changes in the doGet method and create a method for checking palindrome**

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

// **TODO** Auto-generated method stub

String input = request.getParameter("input");

// Check if the input string is a palindrome

boolean isPalindrome = isPalindrome(input);

// Prepare the response

String result = isPalindrome ? "Palindrome" : "Not a Palindrome";

// Send the response back to the user

response.setContentType("text/html");

response.getWriter().println("<html><body>");

response.getWriter().println("<h1>Palindrome Checker</h1>");

response.getWriter().println("<p>The input string is " + result + "</p>");

response.getWriter().println("</body></html>");

}

private boolean isPalindrome(String str) {

str = str.replaceAll("[^a-zA-Z0-9]", "").toLowerCase();

int left = 0;

int right = str.length() - 1;

while (left < right) {

if (str.charAt(left) != str.charAt(right)) {

return false;

}

left++;

right--;

}

return true;

}

**Index1.html**

<!DOCTYPE html>

<html>

<head>

<title>Palindrome Checker</title>

</head>

<body>

<h1>Palindrome Checker</h1>

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

<label for="input">Enter a String: </label>

<input type="text" name="input" required><br>

<input type="submit" value="Check Palindrome">

</form>

</body>

</html>