FirstServlet.java

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

public class FirstServlet extends HttpServlet
{
 public void doGet(HttpServletRequest request,HttpServletResponse response)
 throws IOException,ServletException
 {
  response.setContentType("text/html");
  PrintWriter out=response.getWriter();
  out.println("<html>");
  out.println("<head>");
  out.println("<title>My First Servlet</title>");
  out.println("<body>");
  out.println("<h1>Hello How are U?</h1>");
  out.println("<h2>I am enjoying this Servlet Application</h2>");
  out.println("<h3>See You later!</h3>");
  out.println("</body>");
  out.println("</html>");
 }
}
```

Example 3.7.2 *Write HTML form to read user name and password. This data is sent to the servlet. If the correct user name and password is given then welcome him/her by his/her name otherwise display the message for invalid user.*

Solution :

Step 1 : Create HTML form for accepting user name and password

Input.html

```
<html>
<head>
</head>
<body>
<form action="http://localhost/examples/servlets/servlet/Welcome" method ="get">
User Name:<input type="text" name="uname"/>
<br/>
Password:<input type="password" name="pwd"/>
<input type="submit" value="Submit"/>
</form>
</body>
</html>
```

Step 2 : Create the servlet program to read user name and password and validate it.

Welcome.java

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

public class Welcome extends HttpServlet
{
    public void doGet(HttpServletRequest req,HttpServletResponse res)
    throws ServletException,IOException
    {
        PrintWriter out=res.getWriter();
        res.setContentType("text/html");

        String username=req.getParameter("uname");
        String password=req.getParameter("pwd");
        if ((username=="Ankita")&&(password=="1234"))
            out.print("Welcome "+username);
        else
            out.println("Invalid username");
    }
}
```

Example 3.26.1 *Write AJAX script to obtain the student information stored in XML document. The information should be displayed on clicking the button. It should be displayed in tabular form.*

Solution :

Step 1 : Create an XML file for storing the student information. The XML file is as follows

Student.xml

```
<Student>
    <student_data>
        <Name>AAA</Name>
        <Marks>45</Marks>
    </student_data>
    <student_data>
        <Name>BBB</Name>
        <Marks>55</Marks>
    </student_data>
    <student_data>
        <Name>CCC</Name>
        <Marks>67</Marks>
    </student_data>
    <student_data>
        <Name>DDD</Name>
        <Marks>84</Marks>
    </student_data>
</Student>
```

Step 2 : Create a AJAX script as follows -

AjaxXMLDemo.html

```
<!DOCTYPE html>
<html>
<body>
<h1>STUDENT INFORMATION</h1>
<button type="button" onclick="MyFun()">Click</button>
<br><br>
<table border="1" id="demo"></table>
<script>
function MyFun()
{
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function()
    {
        if (this.readyState == 4 && this.status == 200)
        {
```

```
        Load_XML_File(this);
    }
};
xhttp.open("GET", "Student.xml", true);
xhttp.send();
}
function Load_XML_File(xml)
{
var i;
var xmlDoc = xml.responseXML;
var table="<tr><th>Name</th><th>Marks</th></tr>";
var x = xmlDoc.getElementsByTagName("student_data");
for (i = 0; i <x.length; i++)
{
  table += "<tr><td>" +
  x[i].getElementsByTagName("Name")[0].childNodes[0].nodeValue +
  "</td><td>" +
  x[i].getElementsByTagName("Marks")[0].childNodes[0].nodeValue +
  "</td></tr>";
}
document.getElementById("demo").innerHTML = table;
}
</script>
</body>
</html>
```

Step 3 : Open the web browser and the output will be as follows -

Example 4.5.1 *Write a client server JSP program to find the simple interest and display the result in the client.*

Solution : This program is created using following steps.

Step 1 : Create an simple HTML form for inputting the values of P, N and R. The HTML document for this is as given below.

Input.html

```
<html>
<head><title>Input Form</title></head>
<body>
Enter following values:<br/>
<form method="post" action="interest.jsp">
Amount:<input type="text" name="amount" value="" size="10"/>
Period:<input type="text" name="period" value="" size="3"/>
Rate of Interest:<input type="text" name="rate" value="" size="3"/>
<input type="submit" value="Submit"/>
</form>
</body>
</html>
```

Step 2 : Create JSP page which acts as a client to receive the values for P,N and R. These values will be sent to the server Java program for calculating the simple interest. The calculated interest value is displayed by this same JSP page. The code for this is as follows -

Interest.jsp

```
<html>
<head>
   <title> Simple Interest Calculation Demo </title>
</head>
<body>
<jsp:useBean id="bean_id" class="interestcalcDemo.InterestDemo" scope="session" />
<% String s_p=request.getParameter("amount");
int p=Integer.parseInt(s_p);
    bean_id.setP(p);

String s_n=request.getParameter("period");
int n=Integer.parseInt(s_n);
bean_id.setN(n);

String s_r=request.getParameter("rate");
int r=Integer.parseInt(s_r);
    bean_id.setR(r);
%>
</form>
Amount = <%= p%><br/>
Period = <%= n%><br/>
Rate = <%= r%><br/>
<strong>Interest is now <%= bean_id.getI() %><br/></strong>
</body>
</html>
```

Calling server Java program for computing the simple interest.

Collecting values from **input.html** created in **Step 1** and sending them to sever program using method **set**

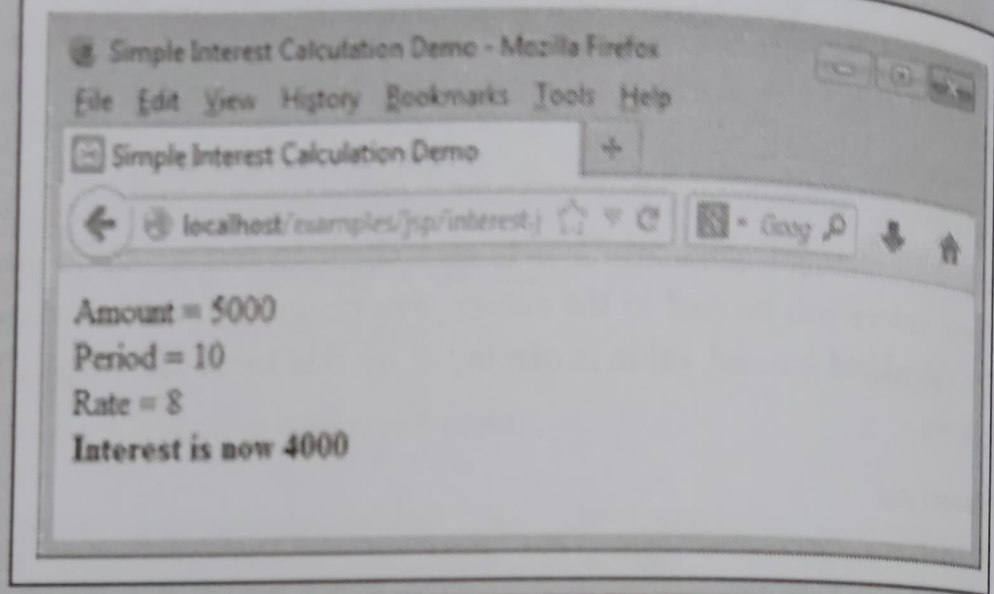
Obtaining interest value from server

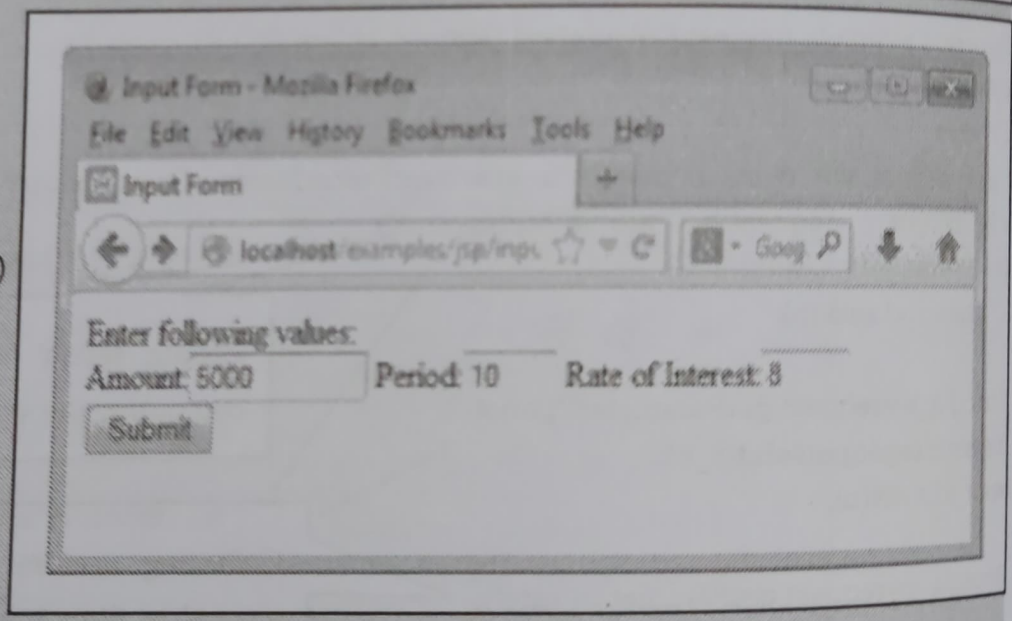
Step 3 : Following is a Java bean program which acts as a Server. It calculated the simple interest using the formula (p*n*r)/100. This value is then returned to the client JSP program created in Step 2. The code for server java program is as follows -

InterestDemo.java

```
package interestcalcDemo;
public class InterestDemo
{
 public int p,n,r;
 public InterestDemo()
 {
     p=0;n=0;r=0;
 }
 public int getI()
 {
     int i=(p*n*r)/100;
     return i;
 }
 public void setP(int p)
 {
     this.p=p;
 }
 public int getP()
 {
     return p;
 }
 public void setN(int n)
 {
     this.n=n;
 }
 public int getN()
 {
     return n;
 }
public void setR(int r)
 {
     this.r=r;
 }
 public int getR()
 {
    return r;
}
```

Output





welcome.html

```
<html>
<head>
<title> FIRST RAILS PROGRAM</title>
</head>
<body>
<center>
<h3>Current Date and Time ... </h3>
<p> It is now <%= t = Time.now %> </p>
</center>
</body>
</html>
```

- Now start the server by going into the test1 directory by the command -

```
>Rails server
```

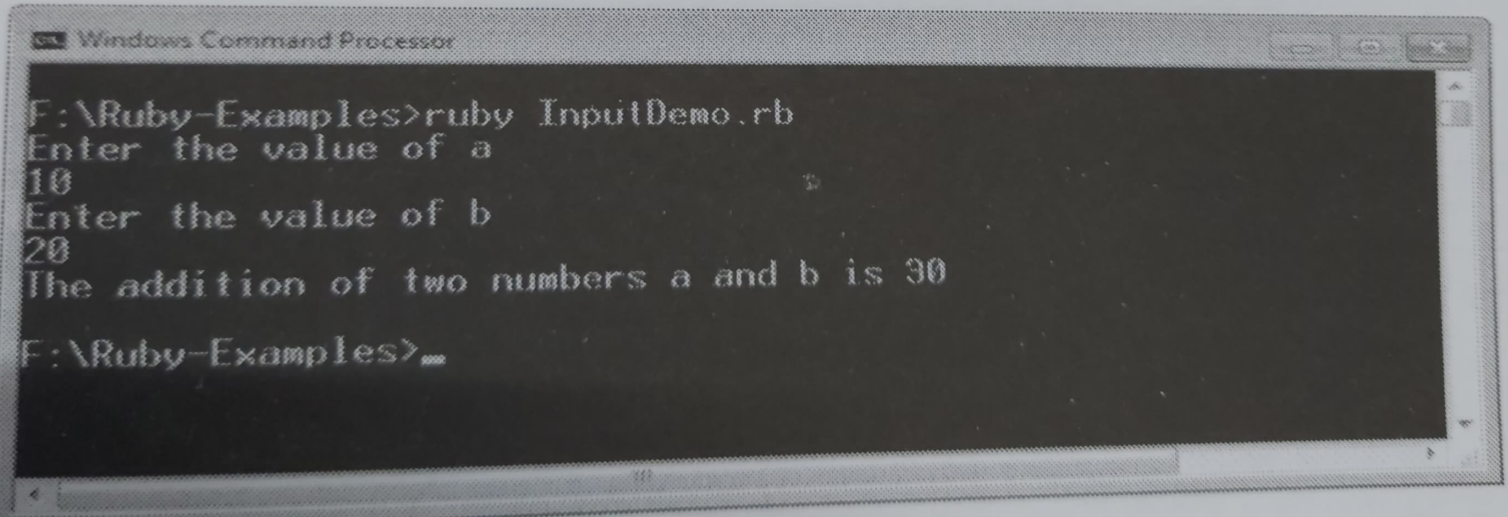
Step 1 : Open the suitable text-editor such as Notepad or Wordpad. Type the following code and save this file using the extension **.rb.**

InputDemo.rb

```
puts "Enter the value of a"
a=gets.to_i
puts "Enter the value of b"
b=gets.to_i
c=a+b
puts "The addition of two numbers a and b is #{c}"
```

Step 2 : Now type the command at the command prompt for executing the ruby program as

Ruby InputDemo.rb



6.5.3 Examples

Following is a simple Ruby program that sorts the array and then searches the desired element from the array.

Ruby Program

```
i=0
A=Array.new
puts "Enter the elements in the array"
while(i<5)
  val=gets
  A[i]=val
  i+=1
end
A.sort!
puts "The sorted array is "
for val in A
  puts val
end
puts "Serching the elements from an array"
puts "Enter the number to be searched"
```

```
key=gets
if A.include?(key)
  puts "The element #{key} is present in the list"
else
  puts "The element #{key} is not present in the list"
end
```

Output(Run1)

```
Enter the elements in the array
30
10
50
40
20
The sorted array is
10
20
30
40
50
Serching the elements from an array
Enter the number to be searched
40
The element 40
 is present in the list
```

(Run1)

```
Enter the elements in the array
30
20
10
50
40
The sorted array is
10
20
30
40
50
Serching the elements from an array

Enter the number to be searched
88
The element 88
 is not present in the list
```