

## Java Applications

1. Using the java programming language we can develop different types of applications, and they are:

**Stand Alone Applications:**

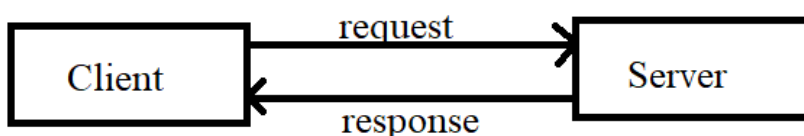
- Standalone applications are also known as desktop applications.
- These applications has to be installed in every machine.
- It can execute only in a single machine where it is installed.
- The installation/uninstallation of application is specific to a client.

**Mobile Applications:**

- An application which is created for mobile devices is called a mobile application.
- Currently, Android and Java ME are used for creating mobile applications.
- The mobile applications can execute only in that platform for which they are developed.

**Web Based Applications:**

- An application that runs on the server side and creates a dynamic page is called a web application.
- These applications are also called as client server applications.
- Currently, Servlet, JSP, Struts, Spring, Hibernate, JSF, etc. technologies are used for creating web applications in Java.



2. Every web based application require 2 types of softwares, and they are: 1) Client Software, 2) Server Software.
3. **Client Software:** It can be any browser that we use in our machine like internet explorer, chrome, firefox etc. Every operating system will provide one default browser.
4. **Server Software:** These servers are available in the market like tomcat, weblogic, glassfish, websphere etc.
5. The user will send the request to the server by using the client software, the server software will receive the request and process the request and then send the response to the client.
6. A web based application consists of web resources (html, images, css, js files etc) and web components (servlets and jsp).
7. To make the communication between client and server in a application we use a protocol called **HTTP**.
8. The web based application that we develop are classified into Static web application and Dynamic web application.
9. **Static:** The Application which won't be changed from person to person and time to time , such type of response is called as Static. Ex: Loginpage, Registrationpage etc. [html]
10. **Dynamic:** The Application which is varied from person to person and time to time, such type of response is called as Dynamic. Ex: Inbox, BankBalance etc. [servlet, jsp etc...]

**Never take your phone out.**

On-board **Google maps navigation.**  
Ather 450X. Quick. Electric. Intelligent.

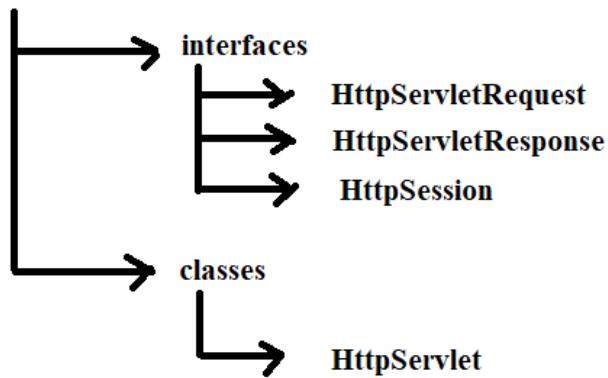
**Book Now**



Google and the G registered trademarks  
Vehicle images shown are for representation

## Servlet

1. Servlet API is a part of JEE API, Servlet is an API for developing web based application.
2. The servlet is a server side technology using which we can develop applications, which run on a server.
3. The servlet API is provided to the programmer in the form of two packages.
  - javax.servlet
  - javax.servlet.http
4. Developing a servlet application means implementing the Servlet Interface either directly or indirectly.
5. A servlet program can be developed in 3 ways :
  - A servlet program can be developed by implementing Servlet Interface.
  - A servlet program can be developed by extending GenericServlet class.
  - A servlet program can be developed by extending HttpServlet class.
6. **javax.servlet package** : Used to develop protocol independent servlets [Generic].
7. **javax.servlet.http package** : Used to develop Http protocol based servlets.



## SPONSORED SEARCHES

[java servlets](#)[servlet tutorial](#)[servlet tomcat](#)[java code](#)[jdbc full course](#)[jsp tutorial](#)

## Servlet Interface

1. The Servlet interface belongs to javax.servlet package and it contains the following 5 methods which has to be implemented.

### Methods of Servlet interface :

- **public void init(ServletConfig)** : This method will be executed only one time, when the servlet object is created or when the first request is sent to the servlet.
- **public void service(ServletRequest, ServletResponse)** : This method is the main method to perform the actual task i.e it will contain the business logic. The servlet container calls the service() method to handle requests coming from the client and to send the response back to the client. The service method checks the HTTP request type (GET, POST, PUT, DELETE, etc) and calls doGet, doPost, doPut, doDelete, etc which is appropriate.
- **public void destroy()** : This method is executed only one time, when the servlet is destroyed.
- **public ServletConfig getServletConfig()** : This method returns an object of ServletConfig.
- **public String getServletInfo()** : This method returns the description about the object.

index.html :

```

-----
<!DOCTYPE html>
<html>
<head>
<title>First Servlet</title>
</head>
<body>
  <form action="FirstServlet">
    <input type="text" name="fname">
    <input type="text" name="sname">
    <input type="submit"/>
  </form>
</body>
</html>

```

## FirstServlet.java

-----

```
package co.in.kothaabhishek;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Date;

import javax.servlet.Servlet;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletResponse;
import javax.servlet.annotation.WebServlet;

@WebServlet("/FirstServlet")
public class FirstServlet implements Servlet{
    ServletConfig config;

    public void init(ServletConfig config) {
        this.config = config;
    }

    public void service(ServletRequest request, ServletResponse response) throws IOException {
        String firstname = request.getParameter("fname");
        String secondname = request.getParameter("sname");

        response.setContentType("text/html");

        PrintWriter pw = response.getWriter();
        pw.println("Hello " + firstname + " " + secondname + " Welcome to our webpage");
    }

    public void destroy() {

    }

    public ServletConfig getServletConfig() {
        return config;
    }

    public String getServletInfo() {
        return null;
    }
}
```

Never take your  
phone out.

On-board Google maps navigation.  
Ather 450X. Quick. Electric. Intelligent.

Book Now



Google and the Go  
registered trademarks of  
Vehicle Images shown in  
are for representation pl

### Explanation :

- An object of **ServletRequest** is used to provide the client request information to a servlet. The servlet container creates a ServletRequest object and passes it as an argument to the servlet's service method.
- **ServletResponse()** Defines an object to assist a servlet in sending a response back to the client. The servlet container creates a ServletResponse object and passes it as an argument to the servlet's service method.
- **getParameter()** : It is used to obtain the value of a parameter by **name**. Returns the value of a request parameter as a **String**, or **null** if the parameter does not exist. You should only use this method when you are sure the parameter has only one value. If the parameter might have more than one value, use **getParameterValues()**.
- **getWriter()** : If you use annotation, deployment descriptor (web.xml file) is not required. **@WebServlet** annotation is used to map the servlet with the specified name. Returns a **PrintWriter** object that can send character text to the client.
- **setContentType()** : Sets the content type of the response being sent to the client, if the response has not been committed yet. The given content type may include a character encoding specification, for example, `text/html; charset=UTF-8`. The response's character encoding is only set from the given content type if this method is called before **getWriter** is called. This method may be called repeatedly to change content type and character encoding.
- Annotations is one new feature introduces in the Servlet 3.0 Specification. Previously to declare servlets, listeners or filters we must do it in the web.xml file. Now, with the new annotations feature we can just annotate servlet classes using the **@WebServlet** annotation..The **@WebServlet** annotation is used to declare a servlet. If you use annotation, deployment descriptor (web.xml file) is not required. **@WebServlet** annotation is used to map the servlet with the specified name. This annotation is processed by the container at deployment time, and the corresponding servlet made available at the specified URL patterns.
- **PrintWriter** : This class gives Prints formatted representations of objects to a text-output stream. It implements all of the print methods found in **PrintStream**. The **PrintWriter** class of the `java.io` package can be used to write output data in a commonly readable form (text).

## GenericServlet Class

1. A **GenericServlet** is a protocol independent Servlet that should always override the **service()** method to handle the client request. You may create a generic servlet by inheriting the **GenericServlet** class and providing the implementation of the service method.

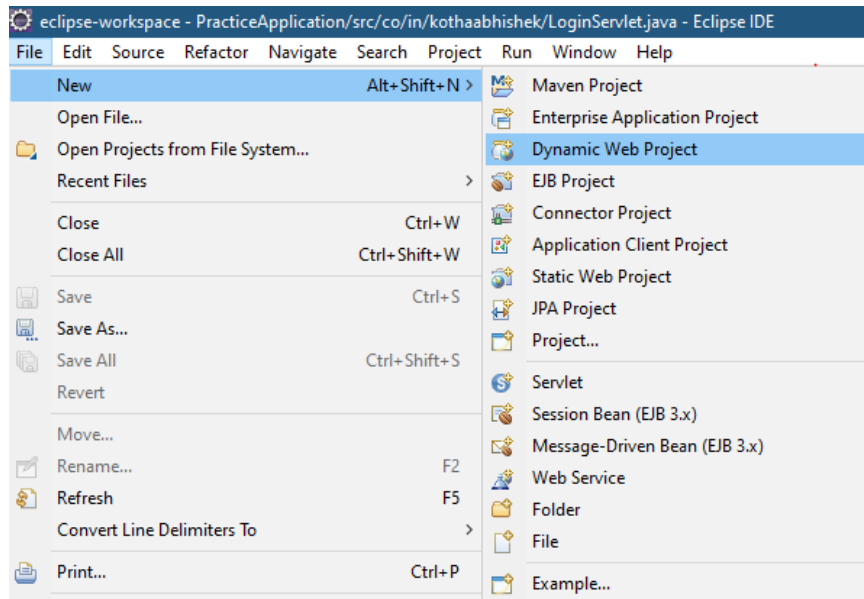
### Methods of Servlet interface to override:

- **public void service(ServletRequest, ServletResponse)** : This method is the main method to perform the actual task i.e it will contain the business logic. The servlet container calls the **service()** method to handle requests coming from the client and to send the response back to the client.

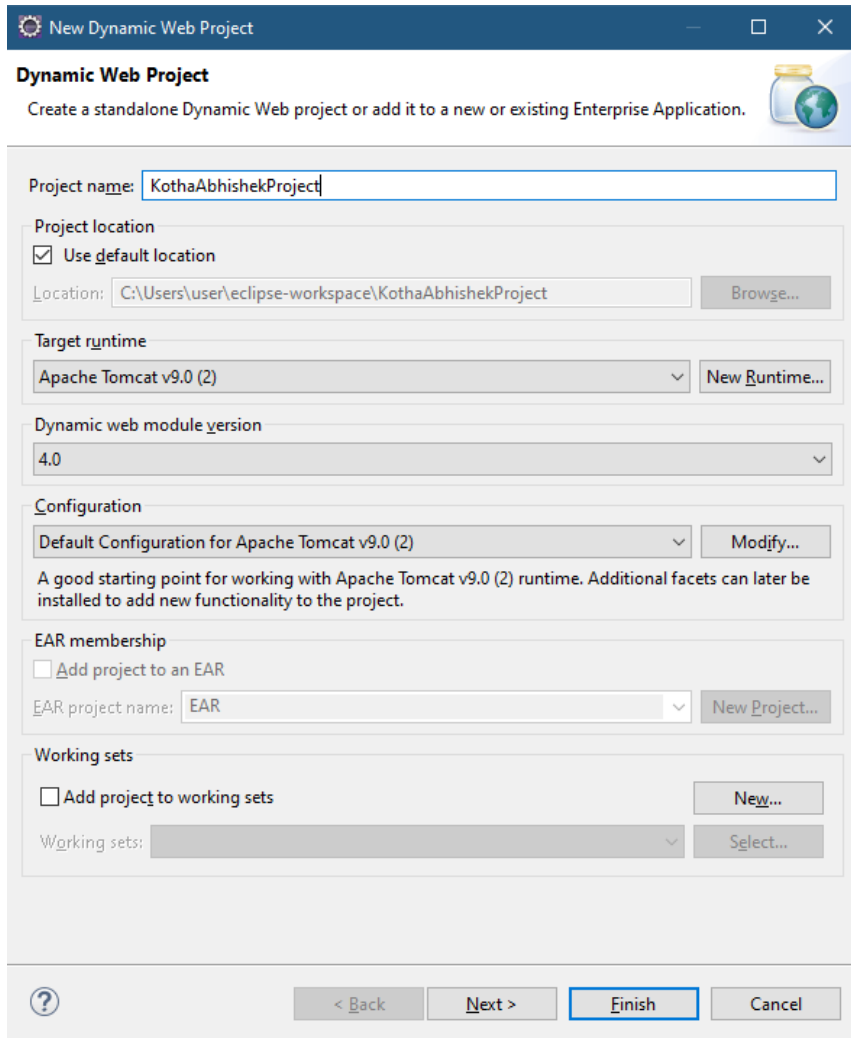


### Procedure to create a GenericServlet using Eclipse :

1. First we have to create a folder with the name of **ProjectName** under that folder we can create all our files like java, html, css etc...
2. After opening the Eclipse first click on **File** and move to **New** and then select **Dynamic Web Project**.

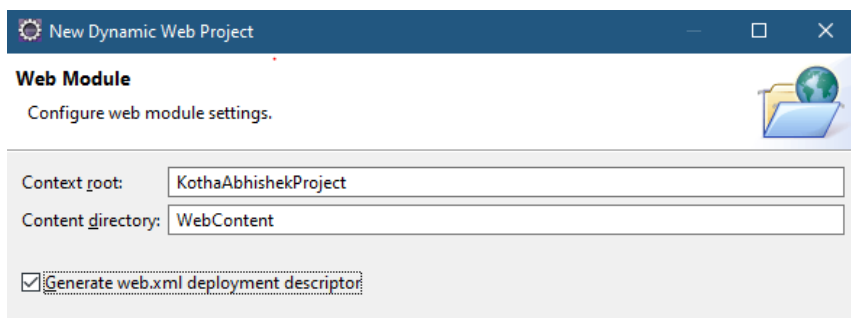


3. In that window First we have to enter our **ProjectName** and then we have to select our **Target Run Time** which means server where we are executing our project. In our case our server is **Apache Tomcat v9.0**, and then select **Dynamic Web Module Version** which is the latest one, in this example 4.0 is the latest one. And click on **next** button 2 times.



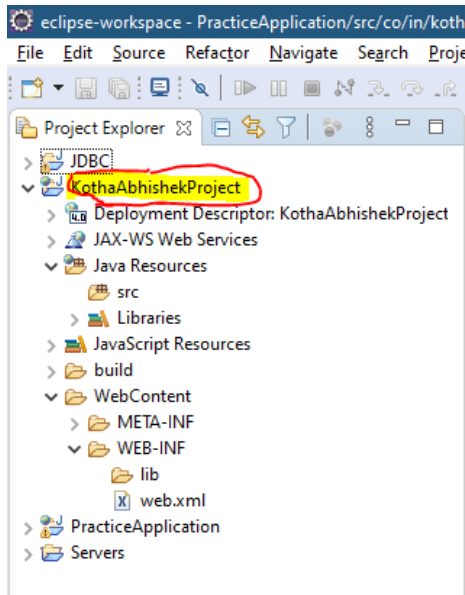
The screenshot shows the 'New Dynamic Web Project' wizard. The title bar says 'New Dynamic Web Project'. The main heading is 'Dynamic Web Project' with a subtitle 'Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.' The wizard is divided into several sections: 'Project name' with a text field containing 'KothaAbhishekProject'; 'Project location' with a checked 'Use default location' checkbox and a text field showing 'C:\Users\user\eclipse-workspace\KothaAbhishekProject'; 'Target runtime' with a dropdown menu set to 'Apache Tomcat v9.0 (2)'; 'Dynamic web module version' with a dropdown menu set to '4.0'; 'Configuration' with a dropdown menu set to 'Default Configuration for Apache Tomcat v9.0 (2)'; 'EAR membership' with an unchecked 'Add project to an EAR' checkbox and an 'EAR project name' field set to 'EAR'; and 'Working sets' with an unchecked 'Add project to working sets' checkbox. At the bottom, there are navigation buttons: '< Back', 'Next >', 'Finish' (highlighted with a blue border), and 'Cancel'.

4. In the next step on the **Web Module** section select **Generate web.xml deployment descriptor**, and then click on **Finish**.

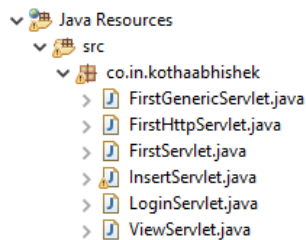


The screenshot shows the 'Web Module' configuration step of the wizard. The title bar says 'New Dynamic Web Project'. The main heading is 'Web Module' with a subtitle 'Configure web module settings.' The wizard has two text fields: 'Context root' set to 'KothaAbhishekProject' and 'Content directory' set to 'WebContent'. Below these fields is a checkbox labeled 'Generate web.xml deployment descriptor' which is checked. At the bottom right, there is a 'Finish' button.

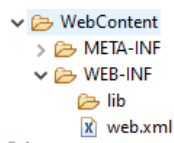
5. On the Left side under **Project Explorer** we can see our Project Folder with the name of our Project.



6. In that Folder we can see many sub folders, in that some folders are very Important. **Java Resources** is the very important folder where we can create our files under **src** folder.

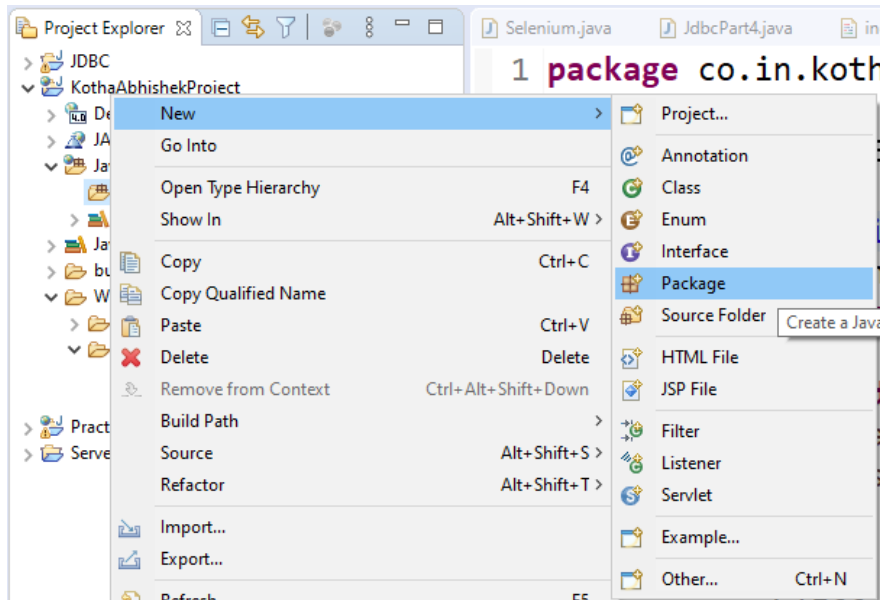


7. Under **WebContent** folder one more important folder is **WEB-INF** folder, in that we have **web.xml** file, with the help of web.xml file only we can call our servlets. By using web.xml file only we can add **welcome-file** which means the webpage will call whenever we execute the program. [For example index.html, default.html, etc...]



```
<welcome-file-list>
  <welcome-file>index.html</welcome-file>
  <welcome-file>index.htm</welcome-file>
  <welcome-file>index.jsp</welcome-file>
  <welcome-file>default.html</welcome-file>
  <welcome-file>default.htm</welcome-file>
  <welcome-file>default.jsp</welcome-file>
</welcome-file-list>
```

8. If you want to create any file like class, Enum, Interface, Package, HTML, jsp, Servlet simply **right click on the project folder move to new and select which file you want to create.**



**Never take your phone out.**

On-board Google maps navigation.  
Ather 450X. Quick. Electric. Intelligent.

[Book Now](#)



Google and its registered trademarks. Vehicle images shown are for representational purposes only.

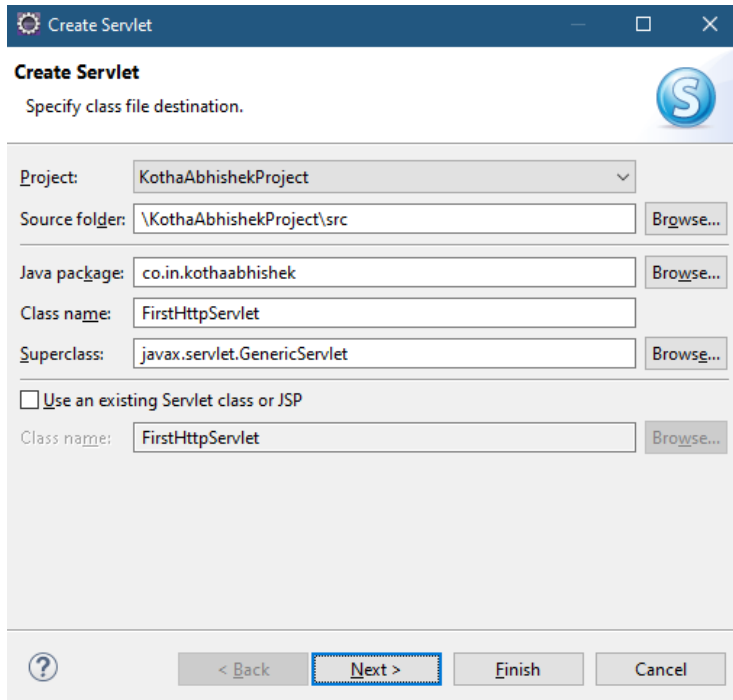
9. In our example first we are creating a HTML file and copy the below code into that html file. While creating html file enter file name as **index.html**, automatically all html files are created under **WebContent** folder only and then click on Finish.

```
index.html
-----
<!DOCTYPE html>
<html>
<head>
<title>First Generic Servlet</title>
</head>
<body>
<form action="FirstGenericServlet" method="get">
  <input type="text" name="fnum"><br>
  <input type="text" name="snum"><br>
  <input type="submit"/>
</form>
</body>
</html>
```

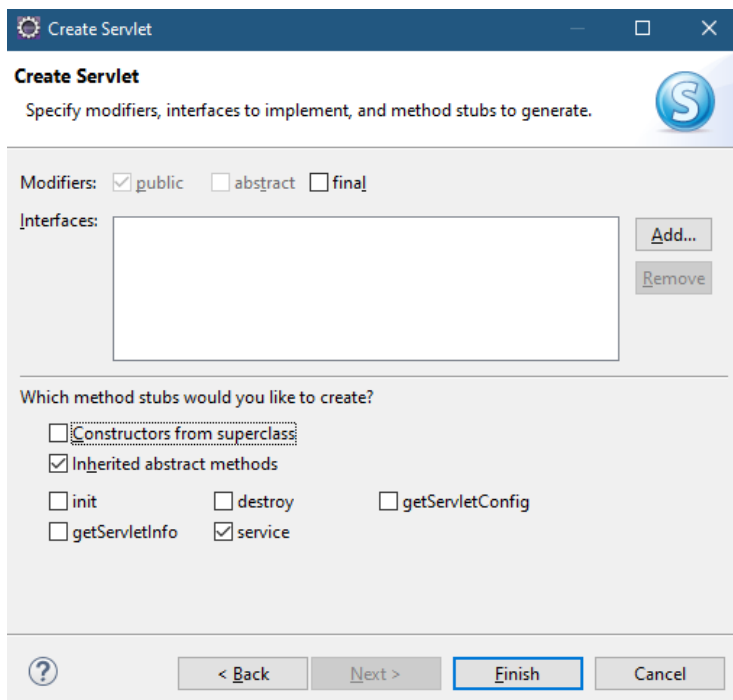
10. In this html file we are creating a Form to accept values from the user. By submitting this form it will call **FirstGenericServlet** because we are using **action** attribute and calling that servlet. In this we are using **method="get"** which means we are submitting values through URL.

localhost:8084/KothaAbhishekProject/FirstGenericServlet?fnum=12&snum=15

11. Create a Servlet with the name of **FirstGenericServlet** and copy the below code into that file. At the time of creating Servlet if you want to create any package enter that package name, here i'm using **co.in.kothaabhishek**, and we have to select super class in this we are creating GenericServlet so we have to select **GenericServlet** as super class by clicking on Browse and search as GenericServlet and select that one. After that click on Next button two times.



12. Next a windows will appear with the name of **Create Servlet** in this we can select which methods we want to override. In this example i want to override **service()** only, so i selected service() and clicked on Finish button.



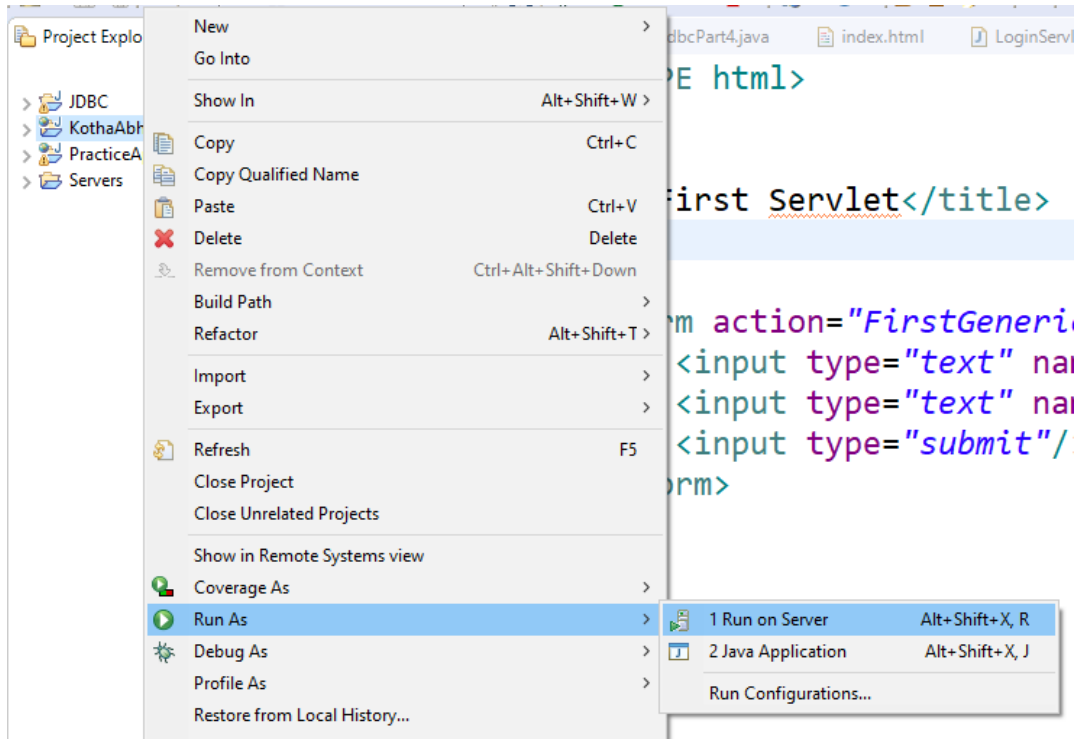
## FirstGenericServlet

```
-----  
package co.in.kothaabishek;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
  
import javax.servlet.GenericServlet;  
import javax.servlet.ServletException;  
import javax.servlet.ServletRequest;  
import javax.servlet.ServletResponse;  
import javax.servlet.annotation.WebServlet;  
  
@WebServlet("/FirstGenericServlet")  
public class FirstGenericServlet extends GenericServlet {  
    private static final long serialVersionUID = 1L;  
  
    public void service(ServletRequest request, ServletResponse response) throws ServletException  
    {  
        int firstnum = Integer.parseInt(request.getParameter("fnum"));  
        int secondnum = Integer.parseInt(request.getParameter("snum"));  
  
        int sum = firstnum + secondnum;  
  
        response.setContentType("text/html");  
        PrintWriter out = response.getWriter();  
        out.print("The sum of two numbers : " +sum);  
    }  
}
```

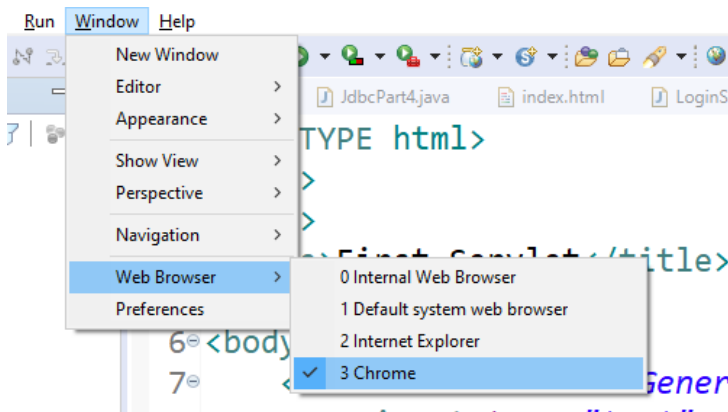
### SPONSORED SEARCHES

[java code](#)[jdbc full course](#)[jsp tutorial](#)[web programming with servlets and jsp](#)

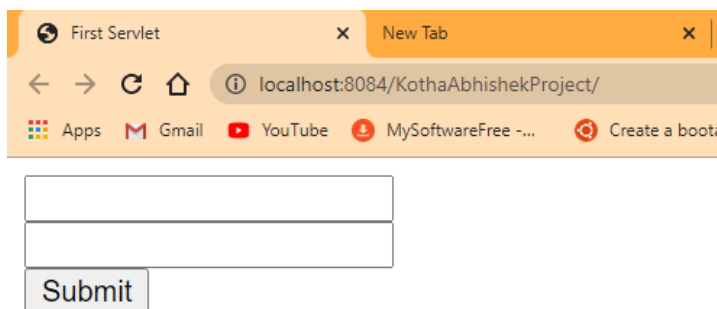
13. We have created a HTML file and sending 2 numbers with the help of Servlet File we are receiving that 2 numbers and we are calculating addition of 2 numbers and printing sum of those numbers.
14. If we want to execute of project first **Right Click** on our project, move to **Run As** and click on **Run on Server**.



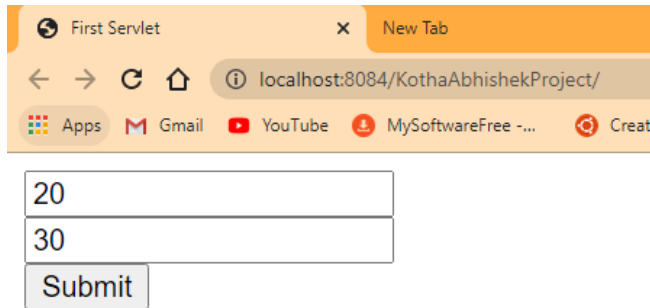
15. On the next window **Run On Server** we have to select **Choose an existing server** and then click on **Always use this server when running this project** and click on Finish. Then the project will execute on the **localhost** and the html page will be displayed on the Internal Browser of Eclipse IDE. If you want to change the default browser click on **Window** tab and move to **Web Browser** and select which browser you want to execute.



16. Again execute by **right clicking** on project folder and move to **run as** and click on **run on server**. It will open the webpage on the browser which you have selected in the last step.



17. Enter any two numbers in text field. And click on submit.



First Servlet x New Tab

localhost:8084/KothaAbhishekProject/

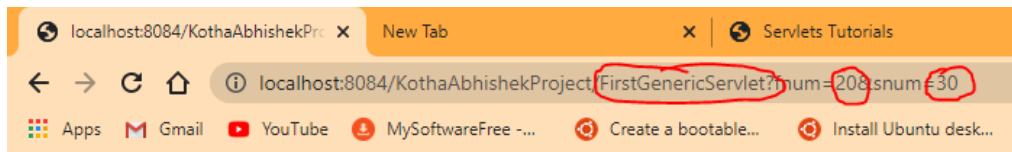
Apps Gmail YouTube MySoftwareFree -... Creat

20

30

Submit

18. The addition of two numbers will be displayed on the web browser with the help of FirstGenericServlet. In this we are using get method so the values are came in URL to the servlet.



The sum of two numbers : 50

19. By default we are getting the values from the **form** in **String** datatype only. But if we want the values in any other datatype like **int**, **double** etc... we will take the help of **Parsing** to convert String format to other datatype format. In our example we have to get **integer** values, so im using parsing technique to convert String datatype to integer.

```
int firstnum = Integer.parseInt(request.getParameter("fnum"));
int secondnum = Integer.parseInt(request.getParameter("snum"));
```

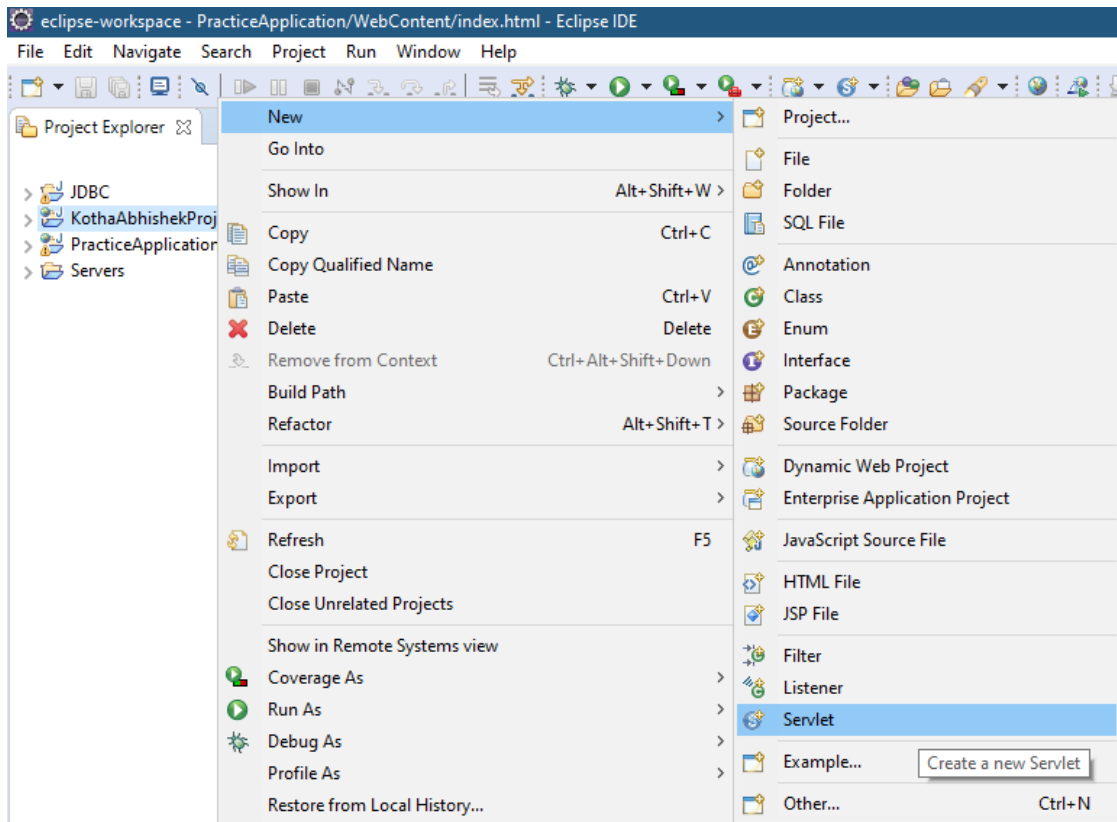


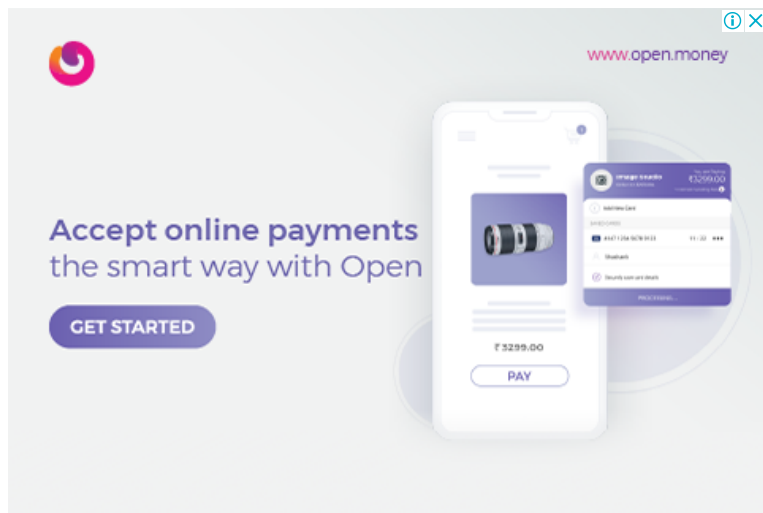
## HttpServlet Class

1. We can create a servlet by extending **HttpServlet** class. This is protocol dependent server, it works only on HTTP protocol.
2. The HttpServlet class extends the GenericServlet class and implements Serializable interface. It provides http specific methods such as **doGet**, **doPost**, **doHead**, **doTrace** etc.
3. Unlike Generic Servlet, the HTTP Servlet doesn't override the service() method. Instead it overrides the **doGet()** method or **doPost()** method or both. The doGet() method is used for **getting the information from server** while the doPost() method is used for **sending information to the server**.
4. If the servlet gets **HTTP GET** request then it will automatically call **doGet()** method, and if the servlet get **HTTP POST** request then it will automatically call **doPost()** method.

### Methods of HttpServlet :

- **protected void doGet(HttpServletRequest request, HttpServletResponse response):** This method is called by servlet service method to handle the HTTP GET request from client. While we are submitting a form by using get method then automatically doGet() method will execute. By using this method we are sending values to server through URL. **Mainly this method will use when we want to get the information from the server.** By using the get method we can send only limited data to the server.
  - **protected void doPost(HttpServletRequest req, HttpServletResponse resp):** This method is called by servlet service method to handle the POST request from client. While we are submitting a form by using post method then automatically doPost() method will execute. **Mainly this is useful when we want to submit the data to the server.** By using this method we can send unlimited data to the server.
  - **protected void doPut(HttpServletRequest req, HttpServletResponse resp):** This method is called by servlet service method to handle the PUT request from client. This method is similar to doPost method but unlike doPost method where we send information to the server, **this method sends file to the server**, this is similar to the FTP operation from client to server.
  - **protected void delete(HttpServletRequest req, HttpServletResponse resp):** Called by servlet service() method to handle the DELETE request from client that **allows a client to delete a document**, webpage or information from the server.
5. Till Last section we learned how to create a ProjectFolder and how to create html or servlet file. But in the Last section we have created GenericServlet but in this section we are creating HttpServlet.
6. First we have to right-click on the ProjectFolder then move to New and click on Servlet..





7. In the next window we have to select Package if you want to create your servlet inside that Package, then we have to enter our Servlet ClassName. And click on Next button 2 times.

A screenshot of the 'Create Servlet' dialog box in an IDE. The 'Project' is set to 'KothaAbhishekProject'. The 'Source folder' is '\KothaAbhishekProject\src'. The 'Java package' is 'co.in.kothaabishek'. The 'Class name' is 'FirstHttpServlet'. The 'Superclass' is 'javax.servlet.http.HttpServlet'. There is an option to 'Use an existing Servlet class or JSP' with a checkbox, which is currently unchecked. The 'Class name' for the existing class is also 'FirstHttpServlet'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish' (highlighted), and 'Cancel'.

8. Now we have to select which methods do you need in your servlet. In this example I'm selecting doGet() and doPost(). And click on Finish button.

A screenshot of the 'Create Servlet' dialog box in an IDE, showing the 'Specify modifiers, interfaces to implement, and method stubs to generate' step. The 'Modifiers' section has checkboxes for 'public' (checked), 'abstract', and 'final'. The 'Interfaces' section has an 'Add...' button and a 'Remove' button. The 'Which method stubs would you like to create?' section has several checkboxes: 'Constructors from superclass' (unchecked), 'Inherited abstract methods' (checked), 'init' (unchecked), 'destroy' (unchecked), 'getServletConfig' (unchecked), 'getServletInfo' (unchecked), 'service' (unchecked), 'doGet' (checked), 'doPost' (checked), 'doPut' (unchecked), 'doDelete' (unchecked), 'doHead' (unchecked), 'doOptions' (unchecked), and 'doTrace' (unchecked). At the bottom, there are buttons for '< Back', 'Next >', 'Finish' (highlighted), and 'Cancel'.

9. Now copy the below code into index.html file.

```
index.html
-----
<!DOCTYPE html>
<html>
<head>
<title>First Servlet</title>
</head>
<body>
  <form action="FirstHttpServlet" method="get">
    <input type="text" name="fnum">

    <input type="text" name="snum">

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

10. Now copy the below code into FirstHttpServlet (created in this section).

## FirstHttpServlet

```
-----  
package co.in.kothaabishek;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
@WebServlet("/FirstHttpServlet")  
public class FirstHttpServlet extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
  
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException {  
        int firstnum = Integer.parseInt(request.getParameter("fnum"));  
        int secondnum = Integer.parseInt(request.getParameter("snum"));  
  
        int sum = firstnum + secondnum;  
  
        response.setContentType("text/html");  
        PrintWriter out = response.getWriter();  
        out.print("The sum of two numbers : " + sum);  
    }  
  
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException {  
        int firstnum = Integer.parseInt(request.getParameter("fnum"));  
        int secondnum = Integer.parseInt(request.getParameter("snum"));  
  
        int sum = firstnum + secondnum;  
  
        PrintWriter out = response.getWriter();  
        out.print("The sum of two numbers : " + sum);  
    }  
}
```

11. In this example we have created doGet() method and doPost() method, if the request is get then doGet() will execute otherwise doPost() will execute.
12. Now rightclick on the project folder and then move to **Run As** and click on **Run On Server**.

The sum of two numbers : 1000

13. In the above example we have created GET method so the values will be sent through URL, But in the next example we are using POST method so the values will be sent through the body. To confirm that check the URL in both the images.

The sum of two numbers : 1000

#### SPONSORED SEARCHES

web programming with servlets and js	software suite	học java web cơ bản
use a cabeça servlets & jsp	java ee	java guide

## Inserting Records into DataBase using HttpServlet

1. If you want to insert records in the DataBase with the help of Servlets, first we have to create a form to accept values from the user.
2. In the second step we have to create a Table to store the values coming from the webpage.
3. In the third step, by using servlet we have to read the values coming from the form and store those values into the DataBase with the help of the JDBC.
4. In this tutorial we are using Oracle Database. First we have to create a table in oracle database to insert values into that table. For that you have to open **sql Plus** command prompt and create a Table with the help of **insert** query. In our example we are creating a **student** table with 3 columns like **sno**, **sname**, **marks**. You can create your own table based on your requirements.

```
Select SQL Plus
SQL> create table student(sno number(5), sname varchar2(20), marks number(3,2));
Table created.
```

5. In the second step you have to create a form with 3 text fields to accept sno, sname and marks from the user and create one button to submit the form. For that purpose create a html file and copy the below code into that html file.

```
index.html
-----
<!DOCTYPE html>
<html>
<head>
<title>First Servlet</title>
<style type="text/css">
  form{
    text-align : center;
  }
  input{
    padding: 5px;
    margin: 5px;
  }
</style>
</head>
<body>
<div>
  <form action="InsertServlet" method="post">
    <input type="text" name="sno"><br>
    <input type="text" name="sname"><br>
    <input type="text" name="marks"><br>
    <input type="submit"/>
  </form>
</div>
</body>
</html>
```

6. In the third step we have to create a Servlet to read the data coming from the user. And we have to store those values into the database so we will take the help of JDBC with in the same servlet. Create a Servlet and copy the below code into that servlet.

InsertServlet:

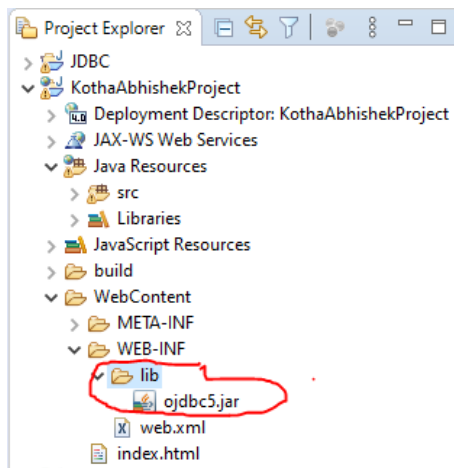
```
-----  
package co.in.kothaabishek;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
import java.sql.Statement;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
@WebServlet("/InsertServlet")  
public class InsertServlet extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
  
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException {  
        int stuno = Integer.parseInt(request.getParameter("sno"));  
        String stuname = request.getParameter("sname");  
        double stumarks = Double.parseDouble(request.getParameter("marks"));  
  
        try {  
            Class.forName("oracle.jdbc.driver.OracleDriver");  
            Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","system", "system");  
            Statement st = con.createStatement();  
            String sql = "insert into student values('"+stuno+"','"+stuname+"','"+stumarks+"')";  
            st.executeUpdate(sql);  
  
            response.setContentType("text/html");  
            PrintWriter out = response.getWriter();  
            out.println("Record is Inserted");  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```



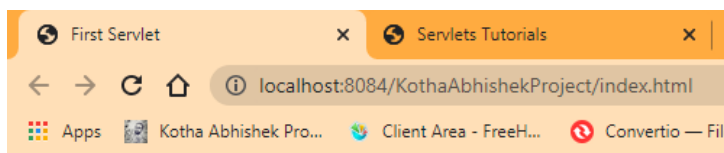
7. All three steps are completed, now we have to execute the program to see the output. For that right-click on Project Folder and move to Run As and click on Run On Server.
8. Then we will get one error like **ClassNotFoundException**, because if we want to connect with the database through Java we have to attach **ojdbc\*.jar** file to that project.

`java.lang.ClassNotFoundException: oracle.jdbc.driver.OracleDriver`

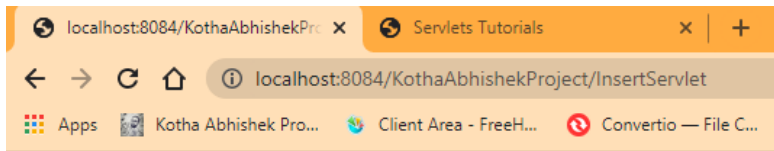
9. Based on your oracle version attach ojdbc.jar (**ojdbc5.jar**, **ojdbc14.jar**, **ojdbc10.jar** etc...) file to the project with in the **lib** folder under **WEB-INF** folder under **WebContent** folder.
10. To add ojdbc file to your project copy ojdbc.jar file and select the lib folder in your project under WebContent and paste (ctrl+v) that file into lib folder.



11. Again execute the project now we will redirect to the webpage in the browser. Enter the values with in the 3 text fields to store into the database and click on submit.

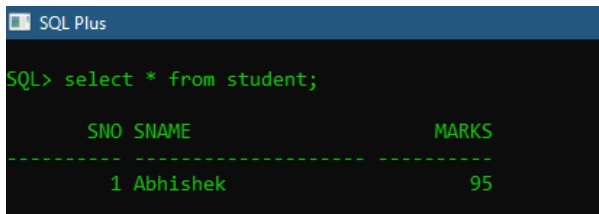


12. Now the record is successfully inserted into the database and we are getting output as **Record is Inserted.**



Record is Inserted

13. If you want to verify the inserted details open **SQL Plus** command prompt and use **select** query to view the records from the student table.



14. You can use this program any number of times to insert records into the student table. In this example we are inserting records into the database with the help of Servlet and JDBC coming from the user through webpage/browser.

### Multiple payment options

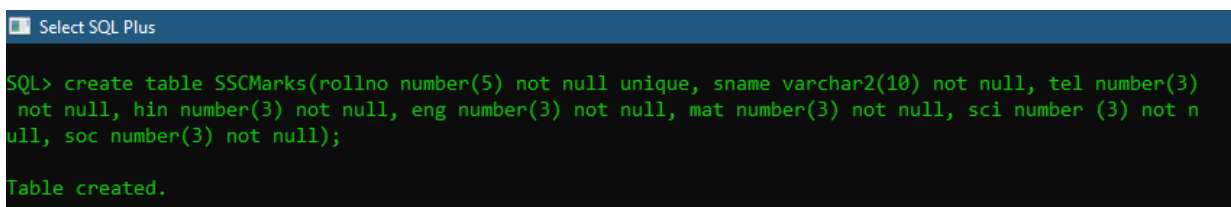
Ad Accept recurring payments. Easy payout option.

Open

Open

## View Records from DataBase using HttpServlet

1. In this example I want to create a Servlet which will get the records from the database, based on a condition. For example after getting results of any exam (like SSC, Inter, BTech) we will enter our HallTicket number in the webpage, based on that number our results will be displayed. In this example we are trying same like that.
2. In the First Step create a table in the database and add some records into that table. In our example I'm creating a table called **sscmarks** with 8 fields to store **sno, sname and student marks in 6 subjects**.



3. Into that table i have added some records (Actually in real time admin/owner will add records into that table user will get the results based on his hallticket number through the browser).

```
SQL> insert into sscmarks values(12345, 'Abhishek', 90, 91, 92, 93, 94, 95);
1 row created.


SQL> insert into sscmarks values(23456, 'Abhilash', 94, 95, 96, 97, 98, 99);
1 row created.

SQL> insert into sscmarks values(34567, 'Vasanth', 44, 55, 66, 77, 88, 99);
1 row created.
```

4. In the Second Step we have to create a form in webpage to accept the HallTicket number from the user. For that purpose we are creating one TextField and one Submit Button. When we click on submit button user has to get his result from the database. Create a html file in your project and paste the below code.

```
<!DOCTYPE html>
<html>
<head>
<title>Retrieve Records in Database</title>
<style type="text/css">
  input{
    padding: 5px;
    margin: 5px;
  }
</style>
</head>
<body>
<div>
<h1>View Records : </h1>
<form action="ViewServlet" method="get">
  <input type="text" name="rollno2" placeholder="Enter Rollno"><br>
  <input type="submit" value="View Result" style="width:170px;"/>
</form>
</div>
</body>
</html>
```

5. In the Third step we have to create a Servlet, and read the value from the browser and based on that number we have to retrieve a record from the database and display that record to the user in browser.

OVHcloud

## Get the best deal on Cloud

Dedicated servers, public cloud, private cloud and more

[Sh](#)

6. If we want to get the information from the table we will use **select** query. For example here im showing that query in **SQL plus** command prompt.

SQL Plus

```
SQL> select * from sscmarks where rollno=12345;
```

ROLLNO	SNAME	TEL	HIN	ENG	MAT	SCI
SOC						
12345	Abhishek	90	91	92	93	94
95						

7. Based on that query we are getting records from the table and we have to display those records in the browser. So create a Servlet and paste the below code into your servlet.



# AMCAT Test for Freshers

Engineering  
Jobs in 2000+  
Companies. Get  
3 Lac inter  
calls every  
month.

AMCAT

Open

View Servlet :

```
-----
package co.in.kothaabishek;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/ViewServlet")
public class ViewServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException {
        int rollno = Integer.parseInt(request.getParameter("rollno2"));

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","system","system");
            Statement st = con.createStatement();

            String query = "select * from sscmarks where rollno="+rollno;

            ResultSet rs = st.executeQuery(query);

            rs.next();

            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            out.println(" Student Rollno : "+rs.getString(1)+"<br>");
            out.println("Student Name : "+rs.getString(2)+"<br>");
            out.println("Telugu : "+rs.getInt(3)+"<br>");
            out.println("Hindi : "+rs.getInt(4)+"<br>");
            out.println("English : "+rs.getInt(5)+"<br>");
            out.println("Maths : "+rs.getInt(6)+"<br>");
            out.println("Science : "+rs.getInt(7)+"<br>");
            out.println("Social : "+rs.getInt(8)+"<br>");

            int total = rs.getInt(3)+rs.getInt(4)+rs.getInt(5)+rs.getInt(6)+rs.getInt(7)+rs.getInt(8);

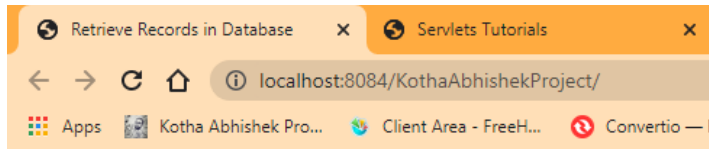
            out.println("Total : "+total);
```

```

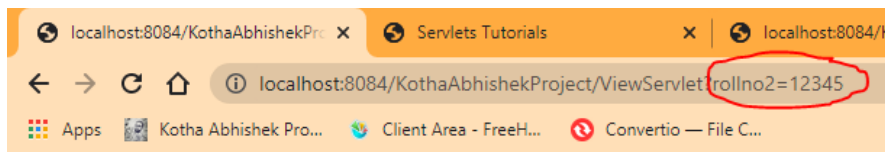
    }catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

8. Now execute the Project by Right-clicking on the folder and move to run as and click on Run on Server. Now the webpage will open in the browser. Now to have to enter your rollno to view your result.



## View Records :

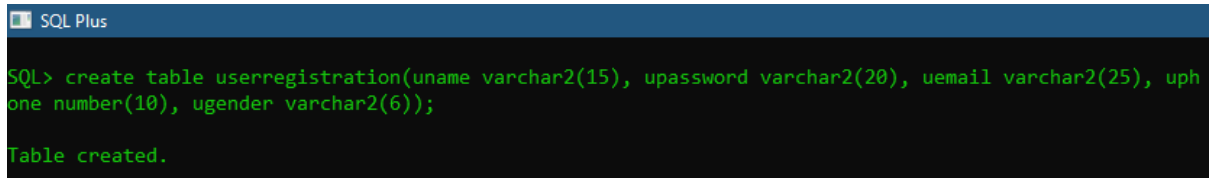
Student Rollno : 12345  
 Student Name : Abhishek  
 Telugu : 90  
 Hindi : 91  
 English : 92  
 Maths : 93  
 Science : 94  
 Social : 95  
 Total : 555

9. Based on the rollno we have retrieved the result from the table and displayed to the user with the help of browser.

SPONSORED SEARCHES

## User Registration & Login using HttpServlet

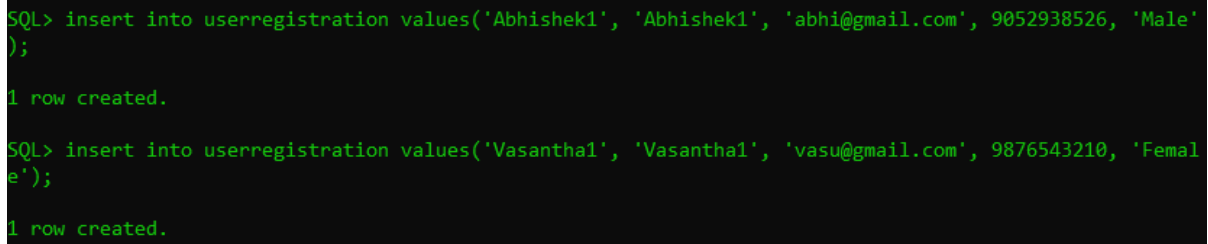
1. In this example we are creating User Registration and Login by using 2 forms with in a single web page. To store the values of User Registration we have created a database table called **UserRegistration** with 5 columns.

A screenshot of a terminal window titled "SQL Plus". The prompt "SQL>" is followed by the command "create table userregistration(uname varchar2(15), upassword varchar2(20), uemail varchar2(25), uphone number(10), ugender varchar2(6));". The response "Table created." is shown on the next line.

```
SQL> create table userregistration(uname varchar2(15), upassword varchar2(20), uemail varchar2(25), uphone number(10), ugender varchar2(6));

Table created.
```

2. I have inserted some records through **SQL plus** command prompt.

A screenshot of a terminal window titled "SQL Plus". The prompt "SQL>" is followed by the command "insert into userregistration values('Abhishek1', 'Abhishek1', 'abhi@gmail.com', 9052938526, 'Male');". The response "1 row created." is shown on the next line. The prompt "SQL>" is followed by the command "insert into userregistration values('Vasantha1', 'Vasantha1', 'vasu@gmail.com', 9876543210, 'Female');". The response "1 row created." is shown on the next line.

```
SQL> insert into userregistration values('Abhishek1', 'Abhishek1', 'abhi@gmail.com', 9052938526, 'Male');

1 row created.

SQL> insert into userregistration values('Vasantha1', 'Vasantha1', 'vasu@gmail.com', 9876543210, 'Female');

1 row created.
```

3. First step is to create a form to Register a user with basic details and create a form to Login with username and Password. For that purpose create an html file and copy the below code into that file.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  font-size: 25px;
}
form{
  width: 35%;
  margin: 0px auto;
}
input[type=text] {
  width: 100%;
  box-sizing: border-box;
  border: 2px solid blue;
  border-radius: 10px;
  font-size: 16px;
  background-color: white;
  padding: 15px 25px 12px 55px;
  margin: 5px;
}
input[type=password] {
  width: 100%;
  box-sizing: border-box;
  border: 2px solid blue;
  border-radius: 10px;
  font-size: 16px;
  background-color: white;
  padding: 15px 25px 12px 55px;
  margin: 5px;
}
input[type=radio] {
  box-sizing: border-box;
  font-size: 16px;
  margin: 20px;
}
input[type=email] {
  width: 100%;
  box-sizing: border-box;
  border: 2px solid blue;
  border-radius: 10px;
  font-size: 16px;
  background-color: white;
  background-size: 25px 25px;
  padding: 15px 25px 12px 55px;
  margin: 5px;
}
input[type=number] {
  width: 100%;
  box-sizing: border-box;
```

```
border: 2px solid blue;
border-radius: 10px;
font-size: 16px;
background-color: white;
padding: 15px 25px 12px 55px;
margin: 5px;
}
input[type=submit] {
width: 100%;
box-sizing: border-box;
border: 2px solid blue;
border-radius: 10px;
font-size: 16px;
background-color: black;
color: white;
padding: 15px 25px 12px 55px;
cursor: pointer;
text-align: center;
}
</style>
</head>
<body>

<form action="UserRegistration" method="post">
  <b>Registration Page : </b><input type="text" name="name" placeholder="Enter Your Name">
  <input type="password" name="pwd" placeholder="Enter Your Password">
  <input type="email" name="email" placeholder="Enter Your email">
  <input type="number" name="number" placeholder="Enter Your Mobile Number">
  Gender :<input type="radio" name="gender" value="Male">Male
        <input type="radio" name="gender" value="Female">Female
  <input type="submit" name="Register" value="Register">
</form>

-----

<form action="ValidateUser" method="post">
  <b>Login Page : </b><input type="text" name="name2" placeholder="Enter Your Name">
  <input type="password" name="pwd2" placeholder="Enter Your Password">
  <input type="submit" name="Login" value="Login">
</form>

</body>
</html>
```

**AMCAT Test for Freshers**

Engineering Jobs in 2000+ Companies. Get 3 Lac interview calls every month.

Ad AMCAT

4. First User will Register with basic details like username, password, email etc.. so we have to store those values into the database table with the help of Servlet and JDBC. Create a servlet to insert values into the table and paste the below code.

```

package co.in.kothaabishek;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/UserRegistration")
public class UserRegistration extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException {
        String name = request.getParameter("name");
        String password = request.getParameter("pwd");
        String email = request.getParameter("email");
        int number = Integer.parseInt(request.getParameter("number"));
        String gender = request.getParameter("gender");

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","system","password");
            Statement st = con.createStatement();

            String query = "insert into userregistration values('"+name+"','"+password+"','"+email+"','"+number+"','"+gender+"')";
            st.executeUpdate(query);

            response.setContentType("text/html");
            PrintWriter out = response.getWriter();
            out.println("User Registered");

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

5. After successful registration user has to login to view the details. So we are creating another servlet to validate user with username & password. If the username and password exists then the user has to view the output, if incase username/password is wrong it has to display invalid user. So create second servlet to verify user credentials and paste the below code in your file.

```
package co.in.kothaabhishek;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/ValidateUser")
public class ValidateUser extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException {
        String name = request.getParameter("name2");
        String password = request.getParameter("pwd2");

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","system","password");
            Statement st = con.createStatement();

            String query = "select * from userregistration where uname='"+name+"' and upassword='"+password+"'";

            ResultSet rs = st.executeQuery(query);

            response.setContentType("text/html");
            PrintWriter out = response.getWriter();

            if(rs.next()) {
                out.println("Name:"+rs.getString(1)+"<br>");
                out.println("Email:"+rs.getString(3)+"<br>");
                out.println("Mobile:"+rs.getString(4)+"<br>");
                out.println("Gender:"+rs.getString(5)+"<br>");
            }else {
                out.println("Not a Valid User");
            }
        }catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

6. Below are some images of User Registration and User Login.

User Registration :

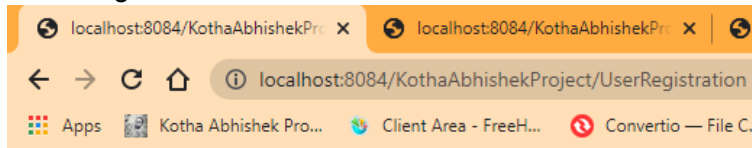
### Registration Page :

Abhishek Reddy
.....
kotha@gmail.com
9052938526

Gender : ☒ Male ☐ Female

Register

User Registration Successful :



User Registered

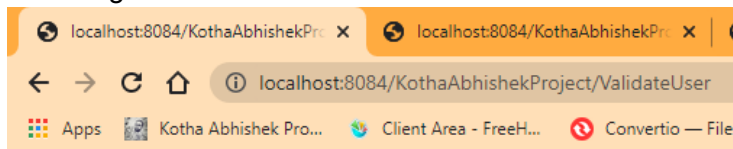
User Login :

### Login Page :

Abhishek1
.....

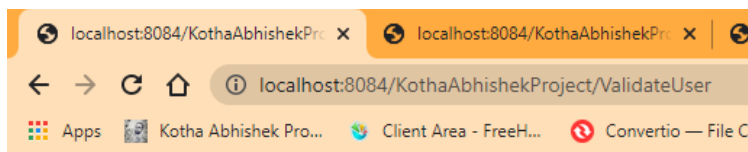
Login

User Login Successful :



Name:Abhishek1  
Email:abhi@gmail.com  
Mobile:9052938526  
Gender:Male

Invalid User :



Not a Valid User

Highly configurable Time Tracking Software  
real-time reports & intuitive dashboards

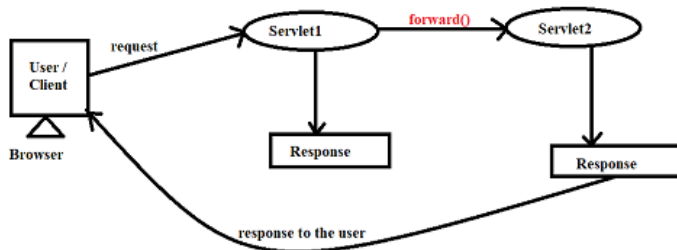
Ad Maximize Productivity & Enable Seamless Digital  
for Employees

ProHance Analytics

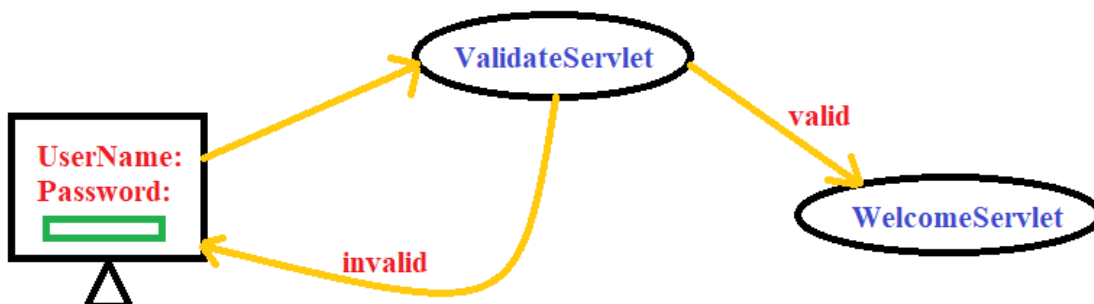
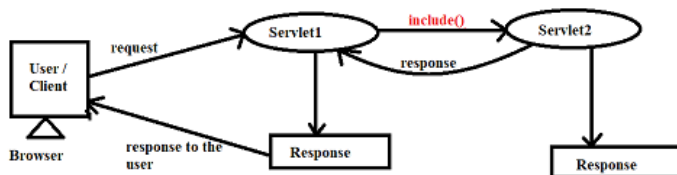
Open

## RequestDispatcher in Servlet

1. The RequestDispatcher interface receives the request from client and dispatches it to the resource it may be html, servlet or jsp.
2. RequestDispatcher interface provides two methods. They are:
3. **forward(ServletRequest request, ServletResponse response)** : Forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server. For example we have 2 Pages Servlet1 and Servlet2, and we are sending request from Servlet1 to Servlet2 by using forward() the final response will send to the client by using Servlet2 page.



4. **include(ServletRequest request, ServletResponse response)** : Includes the content of a resource (servlet, JSP page, or HTML file) in the response. For example we have 2 Pages Servlet1 and Servlet2, and we are sending request from Servlet1 to Servlet2 by using include() the final response will send to the client by using Servlet1 page only.



5. In our example we are creating one html file and 2 servlets. We are accepting username and password through html file, if username and password are valid we have to redirect to WelcomeServlet otherwise we have to print some error message with login page.

index.html :

-----

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
body {
```

```
    font-size: 25px;
```

```
}
```

```
form{
```

```
    width: 35%;
```

```
    margin: 0px auto;
```

```
}
```

```
input[type=text] {
```

```
    width: 100%;
```

```
    box-sizing: border-box;
```

```
    border: 2px solid blue;
```

```
    border-radius: 10px;
```

```
    font-size: 16px;
```

```
    background-color: white;
```

```
    padding: 15px 25px 12px 55px;
```

```
    margin: 5px;
```

```
}
```

```
input[type=password] {
```

```
    width: 100%;
```

```
    box-sizing: border-box;
```

```
    border: 2px solid blue;
```

```
    border-radius: 10px;
```

```
    font-size: 16px;
```

```
    background-color: white;
```

```
    padding: 15px 25px 12px 55px;
```

```
    margin: 5px;
```

```
}
```

```
input[type=submit] {
```

```
    width: 100%;
```

```
    box-sizing: border-box;
```

```
    border: 2px solid blue;
```

```
    border-radius: 10px;
```

```
    font-size: 16px;
```

```
    background-color: black;
```

```
    color: white;
```

```
    padding: 15px 25px 12px 55px;
```

```
    cursor: pointer;
```

```
    text-align: center;
```

```
}
```

```
.error{
```

```
    text-color: red;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
</body>  
<form action="ValidateUser" method="post">  
  <b>Login Page : </b>  
  <input type="text" name="name" placeholder="Enter Your Name">  
  <input type="password" name="pwd" placeholder="Enter Your Password">  
  <input type="submit" name="Login" value="Login">  
</form>  
  
</body>  
</html>
```

ValidateUser.java:

-----

```
package co.in.kothaabhishek;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/ValidateUser")
public class ValidateUser extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException {
        String uname = request.getParameter("name");
        String upassword = request.getParameter("pwd");

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","system", "system");

            String query = "select * from userregistration where uname=? and upassword=?";
            PreparedStatement pst = con.prepareStatement(query);

            pst.setString(1,uname);
            pst.setString(2, upassword);

            ResultSet rs = pst.executeQuery();

            if(rs.next()) { // user valid -----> welcome
                request.setAttribute("output", rs);
                ServletContext ctxt = getServletContext();
                RequestDispatcher rd = ctxt.getRequestDispatcher("/Welcome");
                rd.forward(request, response);
            }else { // invalid user -----> login
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                out.println("Invalid UserName or Password");
                RequestDispatcher rd = request.getRequestDispatcher("/index.html");
                rd.include(request, response);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
    }  
  
    }catch (Exception e) {  
        e.printStackTrace();  
    }  
  
}  
  
}
```

Welcome.java :

-----

```
package co.in.kothaabhishek;
```

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

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

```
import java.sql.ResultSet;
```

```
import javax.servlet.annotation.WebServlet;
```

```
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
@WebServlet("/Welcome")
```

```
public class Welcome extends HttpServlet {
```

```
    private static final long serialVersionUID = 1L;
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException
```

```
    {
```

```
        ResultSet rs = (ResultSet) request.getAttribute("output");
```

```
        PrintWriter out = response.getWriter();
```

```
        out.println("UserName :" + rs.getString(1));
```

```
        out.println("Password :" + rs.getString(2));
```

```
        out.println("Email :" + rs.getString(3));
```

```
        out.println("Phone :" + rs.getInt(4));
```

```
        out.println("Gender :" + rs.getString(5));
```

```
    }catch(Exception e) {
```

```
        e.printStackTrace();
```

```
    }
```

```
}
```

```
}
```

## Get Rid Of Water Leakage

High Quality roof coating protect your terrace from water leakage.



© abhishekkotha96180@gmail.com