

SERVLETS - FORM DATA

<http://www.tutorialspoint.com/servlets/servlets-form-data.htm>

Copyright © tutorialspoint.com

You must have come across many situations when you need to pass some information from your browser to web server and ultimately to your backend program. The browser uses two methods to pass this information to web server. These methods are GET Method and POST Method.

GET method:

The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the ? character as follows:

```
http://www.test.com/hello?key1=value1&key2=value2
```

The GET method is the default method to pass information from browser to web server and it produces a long string that appears in your browser's Location:box. Never use the GET method if you have password or other sensitive information to pass to the server. The GET method has size limitation: only 1024 characters can be in a request string.

This information is passed using QUERY_STRING header and will be accessible through QUERY_STRING environment variable and Servlet handles this type of requests using **doGet** method.

POST method:

A generally more reliable method of passing information to a backend program is the POST method. This packages the information in exactly the same way as GET methods, but instead of sending it as a text string after a ? in the URL it sends it as a separate message. This message comes to the backend program in the form of the standard input which you can parse and use for your processing. Servlet handles this type of requests using **doPost** method.

Reading Form Data using Servlet:

Servlets handles form data parsing automatically using the following methods depending on the situation:

- **getParameter:** You call request.getParameter method to get the value of a form parameter.
- **getParameterValues:** Call this method if the parameter appears more than once and returns multiple values, for example checkbox.
- **getParameterNames:** Call this method if you want a complete list of all parameters in the current request.

GET Method Example Using URL:

Here is a simple URL which will pass two values to HelloForm program using GET method.

`http://localhost:8080/HelloForm?first_name=ZARA&last_name=ALI`

Below is **HelloForm.java** servlet program to handle input given by web browser. We are going to use **getParameter** method which makes it very easy to access passed information:

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class HelloForm extends HttpServlet {

    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
```

```

        throws ServletException, IOException
    {
        // Set response content type
        response.setContentType("text/html");

        PrintWriter out = response.getWriter();
        String title = "Using GET Method to Read Form Data";
        String docType =
            "<!doctype html public \"/>

```

Assuming your environment is setup properly, compile HelloForm.java as follows:

```
$ javac HelloForm.java
```

If everything goes fine, above compilation would produce **HelloForm.class** file. Next you would have to copy this class file in <Tomcat-installation-directory>/webapps/ROOT/WEB-INF/classes and create following entries in **web.xml** file located in <Tomcat-installation-directory>/webapps/ROOT/WEB-INF/

```

<servlet>
    <servlet-name>HelloForm</servlet-name>
    <servlet-class>HelloForm</servlet-class>
</servlet>

<servlet-mapping>
    <servlet-name>HelloForm</servlet-name>
    <url-pattern>/HelloForm</url-pattern>
</servlet-mapping>

```

Now type *http://localhost:8080/HelloForm?first_name=ZARA&last_name=ALI* in your browser's Location:box and make sure you already started tomcat server, before firing above command in the browser. This would generate following result:

USING GET METHOD TO READ FORM DATA

- First Name: ZARA
- Last Name: ALI

GET Method Example Using Form:

Here is a simple example which passes two values using HTML FORM and submit button. We are going to use same Servlet HelloForm to handle this input.

```
<html>
<body>
<form action="HelloForm" method="GET">
First Name: <input type="text" name="first_name">
<br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

Keep this HTML in a file Hello.htm and put it in <Tomcat-installation-directory>/webapps/ROOT directory. When you would access *http://localhost:8080/Hello.htm*, here is the actual output of the above form.

First Name:

Last Name:

Try to enter First Name and Last Name and then click submit button to see the result on your local machine where tomcat is running. Based on the input provided, it will generate similar result as mentioned in the above example.

POST Method Example Using Form:

Let us do little modification in the above servlet, so that it can handle GET as well as POST methods. Below is **HelloForm.java** servlet program to handle input given by web browser using GET or POST methods.

```
// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class HelloForm extends HttpServlet {

    // Method to handle GET method request.
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
        throws ServletException, IOException
    {
        // Set response content type
        response.setContentType("text/html");

        PrintWriter out = response.getWriter();
        String title = "Using GET Method to Read Form Data";
        String docType =
            "<!doctype html public \"-//w3c//dtd html 4.0 \" +
            \"transitional//en\">\n";
        out.println(docType +
            "<html>\n" +
            "<head><title>" + title + "</title></head>\n" +
            "<body bgcolor=\"#f0f0f0\">\n" +
            "<h1 align=\"center\">" + title + "</h1>\n" +
            "<ul>\n" +
            "    <li><b>First Name</b>: "
            + request.getParameter("first_name") + "\n" +
            "    <li><b>Last Name</b>: "
            + request.getParameter("last_name") + "\n" +
            "</ul>\n" +
            "</body></html>");
    }

    // Method to handle POST method request.
    public void doPost(HttpServletRequest request,
```

```

        HttpServletResponse response)
        throws ServletException, IOException {
    doGet(request, response);
}
}

```

Now compile, deploy the above Servlet and test it using Hello.htm with the POST method as follows:

```

<html>
<body>
<form action="HelloForm" method="POST">
First Name: <input type="text" name="first_name">
<br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>

```

Here is the actual output of the above form, Try to enter First and Last Name and then click submit button to see the result on your local machine where tomcat is running.

First Name:

Last Name:

Based on the input provided, it would generate similar result as mentioned in the above examples.

Passing Checkbox Data to Servlet Program

Checkboxes are used when more than one option is required to be selected.

Here is example HTML code, CheckBox.htm, for a form with two checkboxes

```

<html>
<body>
<form action="CheckBox" method="POST" target="_blank">
<input type="checkbox" name="maths" checked="checked" /> Maths
<input type="checkbox" name="physics" /> Physics
<input type="checkbox" name="chemistry" checked="checked" />
Chemistry
<input type="submit" value="Select Subject" />
</form>
</body>
</html>

```

The result of this code is the following form

Maths Physics Chemistry

Below is CheckBox.java servlet program to handle input given by web browser for checkbox button.

```

// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

// Extend HttpServlet class
public class CheckBox extends HttpServlet {

    // Method to handle GET method request.
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException
    {

```

```

// Set response content type
response.setContentType("text/html");

PrintWriter out = response.getWriter();
String title = "Reading Checkbox Data";
String docType =
"!doctype html public "-//w3c//dtd html 4.0 " +
"transitional//en">\n";
out.println(docType +
"<html>\n" +
"<head><title>" + title + "</title></head>\n" +
"<body bgcolor=\"#f0f0f0\">\n" +
"<h1 align=\"center\">" + title + "</h1>\n" +
"<ul>\n" +
"  <li><b>Maths Flag : </b>: "
+ request.getParameter("maths") + "\n" +
"  <li><b>Physics Flag: </b>: "
+ request.getParameter("physics") + "\n" +
"  <li><b>Chemistry Flag: </b>: "
+ request.getParameter("chemistry") + "\n" +
"</ul>\n" +
"</body></html>");
}
// Method to handle POST method request.
public void doPost(HttpServletRequest request,
                    HttpServletResponse response)
    throws ServletException, IOException {
    doGet(request, response);
}
}

```

For the above example, it would display following result:

READING CHECKBOX DATA

- Maths Flag : : on
- Physics Flag: : null
- Chemistry Flag: : on

Reading All Form Parameters:

Following is the generic example which uses **getParameterNames** method of `HttpServletRequest` to read all the available form parameters. This method returns an Enumeration that contains the parameter names in an unspecified order.

Once we have an Enumeration, we can loop down the Enumeration in the standard manner, using *hasMoreElements* method to determine when to stop and using *nextElement* method to get each parameter name.

```

// Import required java libraries
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

```

```

import java.util.*;

// Extend HttpServlet class
public class ReadParams extends HttpServlet {

    // Method to handle GET method request.
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
                      throws ServletException, IOException
    {
        // Set response content type
        response.setContentType("text/html");

        PrintWriter out = response.getWriter();
        String title = "Reading All Form Parameters";
        String docType =
            "<!doctype html public \"-//w3c//dtd html 4.0 \" +
            \"transitional//en\">\n";
        out.println(docType +
            "<html>\n" +
            "<head><title>" + title + "</title></head>\n" +
            "<body bgcolor=\"#f0f0f0\">\n" +
            "<h1 align=\"center\">" + title + "</h1>\n" +
            "<table width=\"100%\" border=\"1\" align=\"center\">\n" +
            "<tr bgcolor=\"#949494\">\n" +
            "<th>Param Name</th><th>Param Value(s)</th>\n"+
            "</tr>\n");

        Enumeration paramNames = request.getParameterNames();

        while(paramNames.hasMoreElements()) {
            String paramName = (String)paramNames.nextElement();
            out.print("<tr><td>" + paramName + "</td>\n<td>");
            String[] paramValues =
                request.getParameterValues(paramName);
            // Read single valued data
            if (paramValues.length == 1) {
                String paramValue = paramValues[0];
                if (paramValue.length() == 0)
                    out.println("<i>No Value</i>");
                else
                    out.println(paramValue);
            } else {
                // Read multiple valued data
                out.println("<ul>");
                for(int i=0; i < paramValues.length; i++) {
                    out.println("<li>" + paramValues[i]);
                }
                out.println("</ul>");
            }
            out.println("</tr>\n</table>\n</body></html>");
        }

        // Method to handle POST method request.
        public void doPost(HttpServletRequest request,
                           HttpServletResponse response)
                           throws ServletException, IOException {
            doGet(request, response);
        }
    }
}

```

Now, try above servlet with the following form:

```

<html>
<body>
<form action="ReadParams" method="POST" target="_blank">
<input type="checkbox" name="maths" checked="checked" /> Maths
<input type="checkbox" name="physics" /> Physics
<input type="checkbox" name="chemistry" checked="checked" /> Chem

```

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

Now calling servlet using above form would generate following result:

READING ALL FORM PARAMETERS

Param Name	Param Value(s)
maths	on
chemistry	on

You can try above servlet to read any other form's data which is having other objects like text box, radio button or drop down box etc.

Loading [MathJax]/jax/output/HTML-CSS/jax.js