Tutorial 5. View Get Post Data

by Dr. Wenjie He

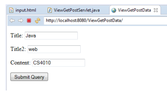
1. Run the example

In this tutorial, we will compare the data sent using the HTTP GET and POST methods. The goal is to help us understand the HTTP protocol better.

You can find the source code of this example in the folder `Tutorial5_ViewGetPostData/Code`. In this folder, there are two sub-folders: `src` and `WebContent`. The `src` folder contains the Java source files, and `WebContent` contains other files for the web application.

The procedure to run the example is similar to that of our tutorial 4 (`Tutorial4_ViewRequestBody`).

We start the project and enter the data as follows,



2. Understand the code

In this example, there are three files: `input.html`, `web.xml`, and `ViewGetPostData.java`. Since the configuration file `web.xml` is very simple and easy to understand, we will look at the other two files.

2.1. Understand the code `input.html`

This file is the welcome file of this web application. It is also used to get the user's input. The user's input data will be sent to the servlet for processing.

Code Listing: `input.html`

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>ViewGetPostData</title>
</head>
<body>
<form method="get" action="viewGetPost">
        <p>Title: <input type="text" name="title"></p>
        <p>Title2: <input type="text" name="title"></p>
        <p>Content: <input type="text" name="content"></p>
        <input type="submit">
</form>
</body>
</html>
```

The first time we use the GET method to send the data. After you click the **Submit Query** button, you will see

You can see that the data is appended the URL line, because the data is sent through the request line.

Then we will use the POST method to send the data. We modify the file by changing the GET method to the POST method.

Code Listing: `input.html`

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>ViewGetPostData</title>
</head>
<body>
<form method="post" action="viewGetPost">
        <p>Title: <input type="text" name="title"></p>
        <p>Title2: <input type="text" name="title"></p>
        <p>Content: <input type="text" name="content"></p>
        <input type="submit">
</form>
</body>
</html>
```

This time you can see that no data is appended the URL line, because the data is sent through the message body.

2.2. Understand the code `ViewGetPostServlet.java`

This is a servlet that we use to process the HTML form submitted from the welcome page `form.html`.

Code Listing: `ViewRequestServlet.java`

```
package edu.umsl.java.web;

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("/viewGetPost")
public class ViewGetPostServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    String title = request.getParameter("title");
    String cont = request.getParameter("content");

    out.println("Title: " + title + "<br />");
    out.println("Content: " + cont + "<br />");

    String[] titleArr = request.getParameterValues("title");

    for (int i = 0; i < titleArr.length; i++) {
      out.println("Title[" + (i + 1) + "]: " + titleArr[i] + "<br />");
    }

    out.flush();
    out.close();
  }

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

}
```

1. (Line 21) To get the value of a parameter, we use the `getParameter` of the `HttpServletRequest` object `request`. This method returns a `String` object. Since all the data from an HTML page is in string format, you get it as a string first, then you can convert it to some suitable data type if necessary.
2. (Line 21) There are two values for the parameter `title`. But when you use the `getParameter` method, only the first value under the name `title` is retrieved.
3. (Line 27) In order to retrieve all the values for the `title`, we need to use the `getParameterValues` method, which returns a string array.
4. (Lines 29-31) We display all the values for the parameter `title` using a for-loop.
5. (Line 39) In the `doPost` method, we simply repeat the logic in the `doGet` method.

2.3. Sending data through the request line using POST method

We know when we send data using an HTML form, the data in the form fields is placed in the message body of the HTTP request. The question is: Can we still send the data through the request line when we use the POST method?

Let us modify the code to see if it is possible.

Code Listing: `input.html`

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>ViewGetPostData</title>
</head>
<body>
<form method="post" action="viewGetPost?pg=2">
        <p>Title: <input type="text" name="title"></p>
        <p>Title2: <input type="text" name="title"></p>
        <p>Content: <input type="text" name="content"></p>
        <input type="submit">
</form>
</body>
</html>
```

Here we add a query parameter `pg=2` after the servlet URL `viewGetPost` to see if this parameter can appear in the request line.

To display the new parameter value in the servlet, we add a few lines in the servlet as follows,

Code Listing: `ViewRequestServlet.java`

```
package edu.umsl.java.web;

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("/viewGetPost")
public class ViewGetPostServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    String title = request.getParameter("title");
    String cont = request.getParameter("content");
    String pg = request.getParameter("pg");

    out.println("Title: " + title + "<br />");
    out.println("Content: " + cont + "<br />");
    out.println("Page: " + pg + "<br />");

    String[] titleArr = request.getParameterValues("title");

    for (int i = 0; i < titleArr.length; i++) {
      out.println("Title[" + (i + 1) + "]: " + titleArr[i] + "<br />");
    }

    out.flush();
    out.close();
  }

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

}
```

When we run the project, we will see

This means that we send the `pg` parameter to the servlet from the request line using the POST method.

==========The End==========