```
Tutorial 4. View Request Body
                                                   by Dr. Wenjie He
1. Run the example
In this tutorial, we will view the structure of a request body through a servlet. The goal of this tutorial is to help us
understand the HTTP protocol and some basic things to process data in a servlet.
You can find the source code of this example in the folder Tutorial 4 ViewRequestBody/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.
Let us first run the example. Then we will understand the details in it.
   1. Start Eclipse for Java EE.
     Next we create a Dynamic Web Project.
  2. In the Eclipse IDE, do the following steps:
        File ⇒ New ⇒ Dynamic Web Project ⇒
      New Dynamic Web Project
       Dynamic Web Project
       Name cannot be empty.
         Project name:
         Project location
         Use default location
         Location: C:\WebTools\IDEs\EclipseEE3\workspace
                                                                     Browse...
         Apache Tomcat v8.5
                                                                  New Runtime...
         Dynamic web module version
         3.1
         Configuration
         Default Configuration for Apache Tomcat v8.5
                                                                     Modify...
         A good starting point for working with Apache Tomcat v8.5 runtime. Additional facets can later
         be installed to add new functionality to the project.
         EAR membership
         Add project to an EAR
         EAR project name: EAR
                                                                   New Project...
         Working sets
          Add project to working sets
                                                                     New...
         Working sets:
                                                                     Select..
         ?
                                < Back
                                            Next >
                                                         Finish
                                                                       Cancel
  3. Type the project name: ViewRequestBody and make sure that Tomcat is selected as your Target
     Runtime.
                                                                         New Dynamic Web Project
       Dynamic Web Project
        Create a standalone Dynamic Web project or add it to a new or existing Enterprise
        Application.
        Project name: ViewRequestBody
         Project location
         Use default location
         Location: C:\WebTools\IDEs\EclipseEE3\workspace\ViewRequestBody
                                                                          Browse...
         Target runtime
         Apache Tomcat v8.5
                                                                        New Runtime...
         Dynamic web module version
         3.1
         Configuration
         Default Configuration for Apache Tomcat v8.5
                                                                          Modify...
         A good starting point for working with Apache Tomcat v8.5 runtime. Additional facets can later be
         installed to add new functionality to the project.
         EAR membership
         Add project to an EAR
                                                                        New Project...
         EAR project name: EAR
         Working sets
         Add project to working sets
                                                                           New...
         Working sets:
                                                                          Select...
        ?
                                                               Finish
                                     < Back
                                                 Next >
                                                                            Cancel
     Then click the Next button.
                                                                 - X
      New Dynamic Web Project
       Java
        Configure project for building a Java application.
        Source folders on build path:
           进 src
                                                               Add Folder...
                                                                 Edit...
                                                                Remove
        Default output folder:
        build\classes
         ?
                          < Back
                                      Next >
                                                    Finish
                                                                 Cancel
     Accept the default setting and click the Next button.
  4. Generate the Deployment Description file for later configuration.
                                                                          _ _ _ X
      New Dynamic Web Project
       Web Module
        Configure web module settings.
        Context root:
                      ViewRequestBody
        Content directory: WebContent
        Generate web.xml deployment descriptor
        ?
                                                              Finish
                                    < Back
                                                Next >
                                                                           Cancel
     In the current window, by default, the Deployment Description checkbox is not selected. We need this file to
     configure our welcome file. Check the checkbox.
                                                                         - - X
      New Dynamic Web Project
       Web Module
        Configure web module settings.
        Context root:
                      ViewRequestBody
        Content directory: WebContent
        Generate web.xml deployment descriptor
        ?
                                                              Finish
                                    < Back
                                                Next >
                                                                           Cancel
     Click the Finish button.
  5. A Dynamic Web Project is created in the Eclipse.
       ViewRequestBody
          ▶ 3 Deployment Descriptor: ViewRequestBody
          JAX-WS Web Services
         🕠 🌁 Java Resources
          JavaScript Resources
          build
         🕟 海 WebContent
     Expand the Java Resources node, the src node will hold all our Java source files.
     Expand the WebContent node. This node will contain all our web resource files.
       ViewRequestBody
          Deployment Descriptor: ViewRequestBody
          JAX-WS Web Services
          Java Resources
           Libraries
          JavaScript Resources
          ⊳ 🗁 <u>build</u>
            WebContent
            META-INF
               WEB-INE.
  6. Copy the Java source files by drag-and-drop.
     The Eclipse IDE supports the drag-and-drop feature to provide the convenience for the programmers.
        Go to the folder Tutorial 4 ViewRequestBody/Code/src and drag the folder edu from it and
     drop it onto the node src that is under the node Java Resources in the project.
       ViewRequestBody
          \mathbb{\frac{1}{200}} Deployment Descriptor: ViewRequestBody
          JAX-WS Web Services
          Java Resources
              D 🖶 edu.umsl.java.web
          JavaScript Resources
          WebContent
  7. Copy the welcome file by drag-and-drop.
        Go to the folder Tutorial 4 ViewRequestBody/Code/WebContent and drag the file
     form.html from it and drop it onto the node WebContent in the project.
     Note: Do not drop it in other sub-folders, such as the WEB-INF folder.

■ ViewRequestBody

          JAX-WS Web Services
          Java Resources

    edu.umsl.java.web

            Libraries
          JavaScript Resources
          build
          WebContent
            META-INF
            form.html
  8. Configure the welcome file inside the web.xml file
     Expand the node WEB-INF and double-click the file web.xml file. Change the filename from
     index.html to form.html. You will see
                     i web.xml ⊠
       form.html
         1 K?xml version="1.0" encoding="UTF-8"?>
         20<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         3 <display-name>ViewRequestBody</display-name>
             <welcome-file>form.html</welcome-file>
               <welcome-file>index.num
               <welcome-file>index.jsp</welcome-file>
               <welcome-file>default.html</welcome-file>
               <welcome-file>default.htm</welcome-file>
               <welcome-file>default.jsp</welcome-file>
        11 </welcome-file-list>
        12 </web-app>
  9. Now we are ready to run the project:
        Rifgt-click the ViewRequestBody project node ⇒ Run As ⇒ Run on Server ⇒
     You will see the page:
       form.html
                     ViewRequestServlet.java
                                            web.xml
                                                         📦 Super Beer 🔀
             http://localhost:8080/ViewRequestBody/
       Select beer characteristics

✓ Body: light
       Color: light
       Can sizes:
        ■ 12 oz.
        ■ 16 oz.

□ 22 oz.

        Submit Query
     Select more than one sizes as follows, so that we can see how to send multiple values for the same
     parameter:
       form.html
                     web.xml
                                                         🚳 Super Beer 💢
             http://localhost:8080/ViewRequestBody/
       Select beer characteristics
       Color: light

✓ Body: light
       Can sizes:

√ 12 oz.

        16 oz.

✓ 22 oz.

         Submit Query
     Click the Submit Query button, you will see the result:
                     web.xml
       form.html
                                                         http://localhost:8080/
                   http://localhost:8080/ViewRequestBody/viewRequest
       127.0.0.1
       Mozilla/5.0 (Windows NT 6.1: Win64: v64: Trident/7.0; rv:11.0) like Gecko
       color=light&body=light@sizes=12oz&sizes=22oz
2. Understand the code
In this example, there are three files: form.html, web.xml, and ViewRequestBody.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 form.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: form.html
      <!DOCTYPE html>
      <html>
     <head>
    <meta charset="ISO-8859-1">
```

2. (Line 8) Here we used the post method, because we want to send the data through the request body. The get method can only send data through the request line. 3. (Line 8) The value for the action attribute is viewRequest, which is the servlet URL pattern (we will

4. (Lines 11-16) We use a dropdown list to send the parameter color. We use the value color for the

6. (Lines 24-26) We use a list of checkboxes to send multiple values for the parameter sizes. Notice that

7. (Line 27) When the submit button is clicked, all the data inside the form will be sent to the server, and

only the data inside this form will be sent. That means the data outside the form will not be sent to the

the three checkboxes use the same parameter name sizes. In this way, when more than one checkboxes

The HTML forms cannot be nested. That is, you cannot put an HTML form inside another HTML form.

An HTML page can contain multiple HTML forms. Each form can have its own back-end processing

• When you have multiple HTML forms in an HTML page, if you click the submit button, only the

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

current form will be submitted to its own back-end data processing file. Data contained in other forms

5. (Lines 18-22) Similarly, we use another dropdown list to send the parameter body.

are checked, multiple values will be sent to the servlet under the same name sizes.

1. (Lines 8-28) This is the main part of this file. It is an HTML form that is used to send the user's data.

<input type="checkbox" name="sizes" value="12oz" />12 oz.

<input type="checkbox" name="sizes" value="16oz" />16 oz.

<input type="checkbox" name="sizes" value="22oz" />22 oz.

<title>Super Beer</title>

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

Select beer characteristics

<select name="color" size="1">

<option>light</option>

<option>amber</option>

<option>brown</option>

<option>dark</option>

<select name="body" size="1">

<option>light</option>

<option>medium</option>

<option>heavy</option>

</select>

<input type="submit" />

see its configuration inside the servlet using an annotation).

name attribute of the select element.

Some basic facts about the HTML forms:

will be ignored.

file placed inside the action attribute.

2.2. Understand the code ViewRequestServlet.java

Code Listing: ViewRequestServlet.java

import java.io.BufferedReader;

package edu.umsl.java.web;

import java.io.IOException;

import java.io.PrintWriter;

@WebServlet("/viewRequest")

Can sizes:

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

</form>

29 </body>

30 </html>

server.

2

3

5

6

12

15

16

17

18

19

20

21

22

23

24

34

35

36

37

38

39

40

41

42

43

44

45

46 }

</head>

<body>

Color:

</select>

Body:

```
import javax.servlet.ServletException;
7
   import javax.servlet.annotation.WebServlet;
   import javax.servlet.http.HttpServlet;
  import javax.servlet.http.HttpServletRequest;
```

HttpServletResponse response) throws ServletException, IOException

import javax.servlet.http.HttpServletResponse;

14 public class ViewRequestServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

protected void doPost(HttpServletRequest request,

String client = request.getHeader("User-Agent");

String ipaddr = request.getRemoteAddr();

response.setContentType("text/html");

out.println(line);

} catch (Exception e) {

out.flush();

out.close();

the client's OS type and the browser's type.

e.printStackTrace();

PrintWriter out = response.getWriter();

```
25
26
         out.println(ipaddr);
         out.println("<br />");
27
         out.println(client);
28
         out.println("<br />");
29
30
         String line = null;
31
32
33
         try {
```

BufferedReader reader = request.getReader();

while ((line = reader.readLine()) != null) {

```
HttpServletResponse object response for this method. We just use them directly in this method.
3. (Line 20) We retrieve the IP address of the user using the method getRemoteAddr(), which could be
  useful for some of the applications.
4. (Line 21) We retrieve the User-Agent information using the method getHeader, so that we can get
```

processing. The servlet container prepares the <code>HttpServletRequest</code> object request and the

1. (Line 13) This is the annotation-type servlet configuration. The URL pattern /viewRequest of this

2. (Lines 17-18) Corresponding to the post method in the HTML form, we use the doPost method for data

servlet is specified in the annotation. This way of configuration is popular due to its simplicity.

content. For different type of response content, we need to choose the appropriate MIME type. 6. (Line 24) Since our response content is of text content, we use the PrintWriter object which is an output stream object for text content.

8. (Lines 34-38) We use a BufferedReader object to print out the message body part line by line.

5. (Line 23) We set the response header Content-Type as the MIME type text/html for the HTML

- 9. (Line 43) When the response content is sent, a buffer is used to carry part of the content. When the buffer is full, then its content will be sent automatically; but if the buffer is not full, its content will not be sent. Now we have completed everything, and we want to force the network to send its content immediately. So we
- call the method flush to do it. 10. (Line 44) The output stream object out occupies memory resource. When we do not need it any more, we should release it, so that we can save some memory resource and make the web application faster.

7. (Lines 26-29) That is the way we produce the text content in a set of string values.

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