

Exercises: Form Data

In this project, you will probably copy and rename your servlet a lot, since later problems are similar to earlier problems. When you do this copying, always change the URL (i.e., the value inside `@WebServlet`). If you have two servlets with the same address, the server will totally ignore one.

1. Open the “forms” project. Run the `ThreeParams` servlet directly via `http://localhost/forms/three-params`. You should see null for all three parameters. Run it again by starting with `three-params-form.html` and submitting the form. You should now see the entries from the textfields.
2. Make a new Eclipse project called `exercises-forms` (or some such). Use this project for the rest of the exercises. Copy the `ThreeParams` servlet from my “forms” project and put it in a new package/directory in your project. Similarly, make a copy of `three-params-form.html`. Test the servlet and the form.
3. Copy/rename the `ThreeParams` servlet and associated form and have the new versions use POST instead of GET. Note that you will have to make small changes to *both* the servlet and the form. You will probably need to restart Tomcat after making the changes (R-click Tomcat and choose Restart).
4. Use `three-params-form.html` to send data to the `ThreeParams` servlet that contains HTML-specific characters. Verify that this can cause malformed results. Make a variation of the servlet that filters the strings before displaying them. Be sure to copy `ServletUtilities` from the “forms” or “test-app” project first, so that you can simply do this:

```
someString = ServletUtilities.filter(someString);
```

5. Make a “registration” form that collects a first name, last name, and email address. Send the data to a servlet that displays it. Feel free to modify the `ThreeParams` HTML form and servlet to accomplish this. Next, modify the servlet to use a default value (or give an error message) if the user omits any of the three required parameters.
6. **[Hard; for the truly inspired.]** Make a variation of the previous servlet and accompanying form, but, in this case, if the user fails to supply a value for any of the fields in the form, you should redisplay the form but with an error message saying that the value is missing. Don’t make the user reenter values that they’ve entered already. Hint: you have two main choices given the tools available to us so far: have one big servlet that generates both the HTML form and the results page (the form submits the data to itself), or two separate servlets (one that generates the HTML form and the other that generates the results page). There is an example of the first approach in the book. If you want to try the second approach, you might want to know about `response.sendRedirect`, which lets you redirect the browser to a specified page. For example, here is a `doGet` method that sends the user to `www.whitehouse.gov`:

```
public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
    throws ServletException, IOException {
    response.sendRedirect("http://www.whitehouse.gov/");
}
```