Assignment 2—Servlets Part 1

Overview

At this point, you have picked a project and implemented a static website with all of the static content that you will need in your final project. In this assignment, you will take the next step by converting your static website into a dynamically generated website and receiving, and well as displaying data from forms. If you have not completed all of the requirements of assignment 1, you are encouraged to do so before beginning assignment 2, as assignment 2 directly builds upon assignment 1 (except for the javascript). As always, you may change your mind about the final project and choose a different option, but the assignment 1 content for the other option will need to be implemented before beginning assignment 2.

Assignment 2 Requirements

For assignment 2, all pages of your project should be placed into servlets and generated dynamically. This means that they will no longer exist as html files, and will instead be "spit out" line by line by a servlet. Obviously, the most straightforward way to do this is to copy/paste your html content into a servlet and add the code to spit out this content.

As an example, if a snippet of an html page looks like this

```
<h1>Big Header Here</h1>
<p>This is a paragraph, content about a website is here</p>
```

it would instead be produced by (in most cases) the doGet method of the appropriate servlet and would likely look something like this

```
out.println("<h1>Big Header Here</h1>");
out.println("<p>This is a paragraph, content about a website is here</p>");
```

In addition, the servlets should receive data from forms and output this data instead of the webpage when the data is sent through a form. When this data is output, do not worry at all about style. Simply perform any necessary calculations mentioned below and output the data to a blank webpage. We will work with this data in future assignments. As an example, suppose you have the following form

```
<form action = "mywebpage" method="post">
<input type="text" name = "temperature" value="freezing"></input>
</form>
```

When the webpage is displayed normally, you should see the webpage with this form. However, if the user clicks the button and sends the value “freezing” to the servlet, the servlet should receive this and instead generate a totally blank page that contains the word “freezing” In this example, simply the word “freezing” is sufficient. No other tags or content of any kind would be required on this page. Detailed requirements are provided on a page by page basis below.

1. User account registration page

The user account creation page should display a blank page containing all of the data entered by the user when the user clicks the create account button. If some of the fields are left blank, the servlet should not crash and should display the other data.

2. Home/Login page

The login page should display a blank page containing the username and password entered when the login button is clicked. If some of the fields are left blank, the servlet should not crash and should display the other data.

3. Content Page 1

For the stock exchange option:

When the user clicks a “buy” button beside a stock and enters a quantity in the quantity field, the servlet should display a blank page containing the word “buy”, the stock name, the number of shares attempting to be purchased, and the total cost of those shares. When the sell button beside a stock is clicked, the servlet should display display a blank page containing the word “sold”, the stock name, the number of shares attempting to be sold, and the total value of those shares. The servlet should not crash when a quantity is not provided when either buying or selling, and should display the data that it has received.

For the online marketplace option:

When “add to cart” is clicked for an item, the servlet should generate a blank webpage containing the name of the item, the quantity chosen, and total price. The servlet should not crash when a quantity is not provided and should display the data that it has received.

4. Content page 2.

For the stock exchange option:

When the user clicks the “add money” button, the servlet should generate a blank page that says “added funds” and the total amount of money entered into the field. It should not crash if no amount has been provided and should display the data it has received.

For the online marketplace option:

When the “clear cart” button is clicked, the servlet should generate a blank webpage that says “clear cart.” Likewise, when the checkout button is clicked, the servlet should display a blank webpage that says “checkout.” When the user clicks the “add money” button, the servlet should generate a blank page that says “added funds” and the total amount of money entered into the field. It should not crash if no amount has been provided and should display the data it has received.

5. Content page 3. (if online marketplace option is chosen)
If you have selected the online marketplace option, content page 3 must be converted into a servlet like the other pages. However, no additional functionality beyond this is required on this page for assignment 2.

If you proposed an independent project and it was approved, in this assignment, you must fully convert your application into servlets. All servlets must receive all data from your forms and process that data as necessary. They should then output this processed data in the form of a blank webpage.

The site should use css to implement some sort of simple color scheme and be reasonably aesthetically pleasing. (Basically it should not look 'tacky').

All pages should be linked to all the other pages. Links should be in a consistent position. They should not move or break from page to page.

NOTICE: SUBMISSION INSTRUCTIONS HAVE CHANGED

If you are using netbeans, please make sure that you correctly **EXPORT TO ZIP** within netbeans as shown in class. This means, within netbeans, click “file” at the top left, then navigate to “Export Project” and choose “To Zip...” This will open a dialog box. Select your project and click the “Export” button. Do not simply zip the directory itself with windows or your project will not load into my netbeans installation correctly. **If it does not load into my netbeans installation correctly, you will get a 0 on your assignment.** To test whether or not you are doing it correctly, please backup your project, export it from netbeans, delete it from netbeans, and import your exported copy. As with the last assignment, you do not have to adhere to any naming convention, but if you would like to, you may name your project NetbeansLastname where Lastname is your last name. Please upload this zip file into moodle.

If you are using eclipse, the instructions have also changed. The zip files do not load properly into netbeans so I will be grading your applications with a separate eclipse installation. Eclipse has the ability to easily export war files with the source code. **Please export a war file of your project** by right clicking your project within eclipse and navigating to the option called “Export.” Choose “war file” and a dialog will appear. Check the box for “export source files” and click finish. Save your file somewhere. Rename your file “EELastname” where Lastname is your last name. Please upload this file into moodle. Please adhere to this naming scheme so I know to use eclipse to load your application. Do not simply zip the directory itself with windows or your project will not load into my eclipse installation correctly. **If it does not load into my eclipse installation correctly, you will get a 0 on your assignment.** To test whether or not you are doing it correctly, please backup your project, export it from eclipse, delete it from eclipse, and import your exported copy.

For both netbeans and eclipse, provide a url for your application's home page or login page in the comments section in moodle. Unlike 2300, this will be a local url. It will probably look like one of the following

http://127.0.0.1/yourpackage/yourservlet

or

http://localhost/yourpackage/yourservlet

but definitely not

C://users/name/workspace/somefile.html

Please do not attempt to use notepad++ or dreamweaver for this assignment. They both work great for simple web development but do not support dynamic web development at all.

Grading Rubric

10 points -All site content present and site links working and in consistent position. Submission link works and submission directions followed.
10 points- Css styles implemented. Site has color scheme and has a good look and feel.
40 points- All pages are generated from servlets instead of existing as static html files.
30 points- Servlets receive appropriate form data and output appropriate data.
10 points- Servlets do not crash when some form data is not provided to them, and instead display the data they have received.

Total 100 points