

CPRG352 - Web Application Programming

Fall 2021

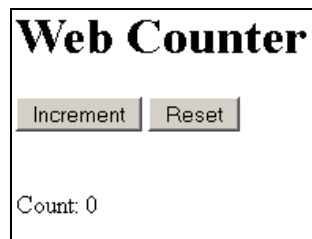
Topic: State Management

TO FOCUS ON THE STATE-MANAGEMENT ASPECT OF THESE PROBLEMS THE PROBLEM SPECIFICATIONS BELOW USE SERVLETS TO IMPLEMENT ALL THE FUNCTIONALITY IN THE PROBLEM APPLICATIONS. YOU MAY ALSO USE A COMBINATION OF SERVLETS AND JSPs IN YOUR SOLUTIONS IF YOU SO WISH

Problem 1: Storing State using a hidden field in a HTML form

Create a web application called **WebCounter** that lets a user increment a number, or reset the count to zero.

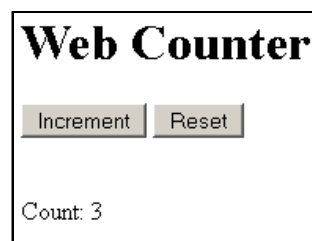
When first run the application should show the following form:



The image shows a web form titled "Web Counter". It contains two buttons, "Increment" and "Reset", and a text label "Count: 0".

Clicking on the **Increment** button adds one to the count value each time it is clicked.

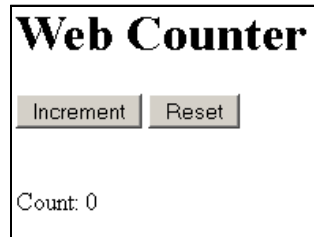
Example: the **Increment** button was clicked three times by the user



The image shows the same web form titled "Web Counter" with buttons "Increment" and "Reset", but the text label now displays "Count: 3".

Clicking on the **Reset** button resets the count to zero.

Example: User clicked on the **Reset** button



The image shows a web application interface titled "Web Counter". It contains two buttons, "Increment" and "Reset", and a text label "Count: 0".

Application Requirement:

The count value must be stored in a hidden HTML form field in the page shown. The **Increment** and **Reset** buttons are form submit buttons that submit the form to a servlet. The servlet takes the current count value, increments it, and outputs a new version of the HTML page with the new count hidden in it, and also displayed as shown above (if the user clicked on the **Increment** button). The servlet resets the count value to 0 if the **Reset** button was clicked, and again outputs a new copy of the page as above.

Hint: if you have more than one submit button in the same HTML form and you give each button a different name, then you can check in the parameters received by a servlet to see which button was clicked (look for a parameter with the same name as the name of the button selected).

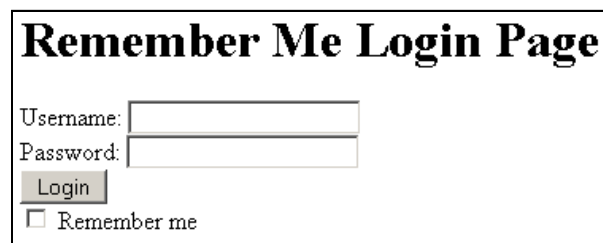
Problem 2: Storing State using Cookies

Note: this problem is very similar to one that we implemented in the lecture. Try to implement a solution to this problem yourself, without referencing that project (if you can then you understand this topic 😊).

Create an application called **RememberMe**. This application will have two servlets:

1. **LoginPage**: displays a simple login form to the user which includes an option to “remember” the username (so the user doesn’t have to re-enter it every time s/he logs in)
2. **Validate**: a servlet which compares the username and password with a known user (username of “*user*”, password of “*pass*”) and sends the user back to LoginPage to display a message (see below)

When run the application should show the **LoginPage** servlet first:

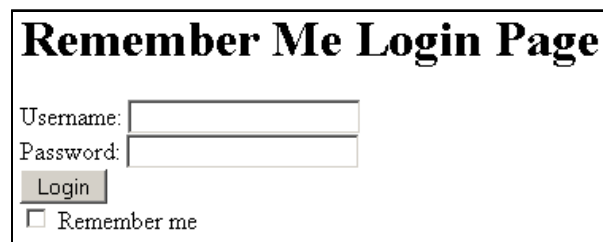


The screenshot shows a web form titled "Remember Me Login Page". It contains two input fields: "Username:" and "Password:". Below the "Password:" field is a "Login" button. At the bottom of the form is a checkbox labeled "Remember me".

Under the **Login** button there is a “**Remember me**” check box which, when checked, tells the application the user wants it to automatically fill in (remember) the username field of the logging-in user in future (to save the user some typing). This application must use a *cookie* to hold this value.

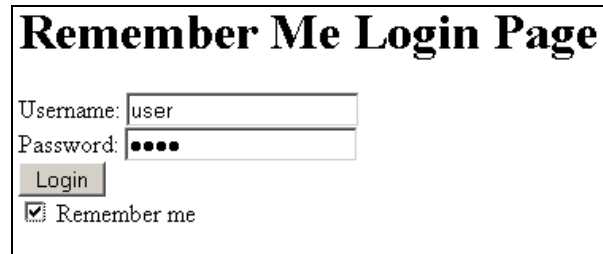
Sample Run:

- 1) User runs application, “**Remember me**” not yet set by the user.



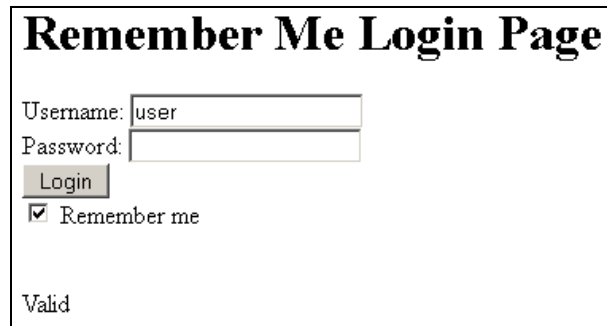
This screenshot is identical to the one above, showing the "Remember Me Login Page" with the "Username:" and "Password:" fields, the "Login" button, and the unchecked "Remember me" checkbox.

- 2) User fills in his/her details, and checks “**Remember me**” check box



The screenshot shows a web form titled "Remember Me Login Page". It contains two input fields: "Username:" with the value "user" and "Password:" with four black dots. Below the password field is a "Login" button. At the bottom of the form is a checked checkbox labeled "Remember me".

- 3) User clicks on the **Login** button, form is submitted to **Validate** servlet which validates the username/password (against “user” and “pass”) and then sends the user’s browser back to LoginPage to display the result (“Valid” or “Invalid”)



The screenshot shows the same "Remember Me Login Page" form. The "Username:" field is still filled with "user", but the "Password:" field is now empty. The "Login" button is still present. The "Remember me" checkbox is still checked. At the bottom of the form, the word "Valid" is displayed.

The “**Valid**” message at the bottom of this page just tells us that the user entered a known username (“user”) and password (“pass”). The **Username** text box is now pre-filled with the username, so user does not have to enter that data any more. We will not pre-fill the **Password** box (for obvious reasons)!

Also, the “**Remember me**” check box is pre-checked, because we are choosing to remember the username at this time.

Only usernames for known (validated) users will be “remembered” this way. Invalid usernames (ones not validated by **Validate**) should not be remembered.

- 4) User enters an invalid password, and attempts to log in again (message “**Invalid**” is displayed below form)

Remember Me Login Page

Username:

Password:

☒ Remember me

Invalid

The message “**Invalid**” is displayed, telling the user than s/he entered an invalid username and/or password. Otherwise the output is as expected.

- 5) User clears “**Remember me**” check box and enters a valid password

Remember Me Login Page

Username:

Password:

☐ Remember me

Invalid

Note: The “**Invalid**” message below the login form is still displayed because of what the user did in step 4.

- 6) User clicks on **Login** button

Remember Me Login Page

Username:

Password:

☐ Remember me

Valid

Note that this time the “**Valid**” message is still displayed, but the username not being pre-filled into the **Username** text box, and the “**Remember me**” check box is clear.

State Management:

The username value must be stored in a cookie so it can be stored for each user on his/her own computer. The **Validate** servlet will check for three parameters coming from the login form:

- username
- password
- rememberme

These are the three HTML controls that should exist in the login form. The **Login** button is the submit button for the form.

How will the cookie data be used?

LoginPage will check for cookie data for a cookie containing a username to remember . If found this tells **LoginPage** to show the username as the value of the username HTML control, and to check the "***Remember me***" checkbox automatically. If no username is available then the username HTML control should be empty, and the checkbox should be unchecked by default.

Validate should check to see if the **rememberme** parameter was received. If it was (the **rememberme** parameter is not *null*) then a new cookie should be created to hold the associated **username** value, and this cookie should be sent back to the browser with the next response. If the **rememberme** parameter is *null* then delete the cookie on the user's computer containing their username because the user has not chosen to have the system remember it.