Jonathan Tweedle

2019/07/31

CSHP 230 B Su 19: Web Applications In C#

Assignment 06

Managing State Data

# Introduction

For this assignment the goal is to show three different techniques for storing and retrieving data between page views. This is generally known as managing page state data and while there are many ways to achieve this, I plan to demonstrate three of the more commonly used techniques:

1. Hidden Input Fields
2. Session Data
3. Cookie Data

# Getting Started

I started with a blank ASP.Net web application as in the previous assignments. While not needed, imported Bootstrap using the NuGet package manager to make the design look moderately professional.

I created a simple navigation menu on the master page to help move between each of the demonstration pages with shell pages to handle each of the demonstrations (Figure 1).

I can now start building out each of the demonstrations.

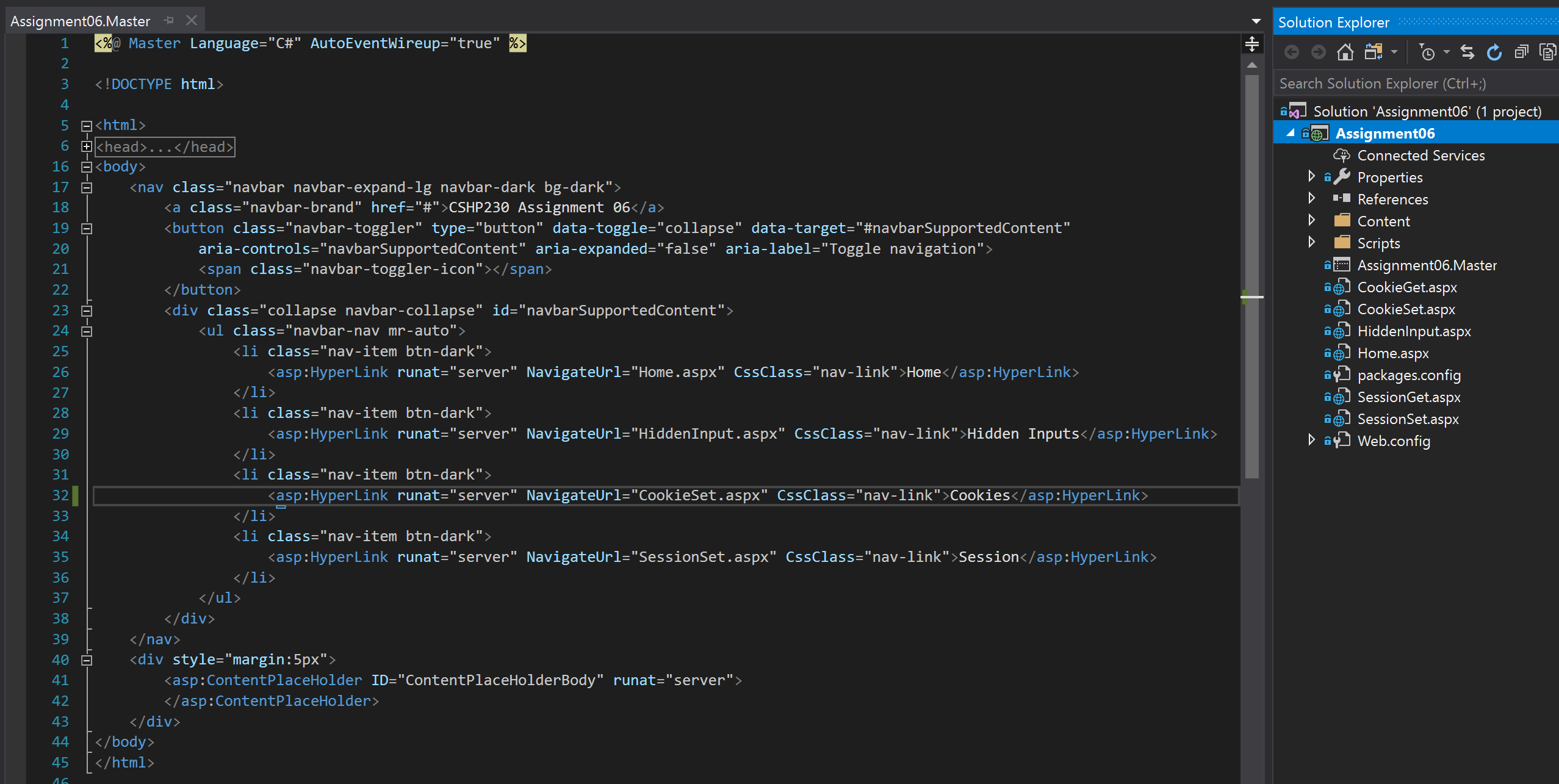


Figure 1: Master page and starting layout of the site

## Hidden Input Demonstration

Forms can make use of hidden input fields to store data that will then get sent to the server when the page is posted. The server can then read the values as if it was any normal control.

For this demonstration I used a simple form with two buttons. The first one triggers a JavaScript function (Figure 2) which gets the current value from the HiddenField which happens to be a simple counter and increases the value by one, the result is then stored back into the HiddenField.

When the other button is clicked, the form is posted. When the page is loaded as a result of the post call, the Page\_Load function (Figure 3) obtains the value from the hidden field. Using the value it updates a label to inform the user how many times the button was clicked.

I also included a Click handler to show how this could be triggered here.

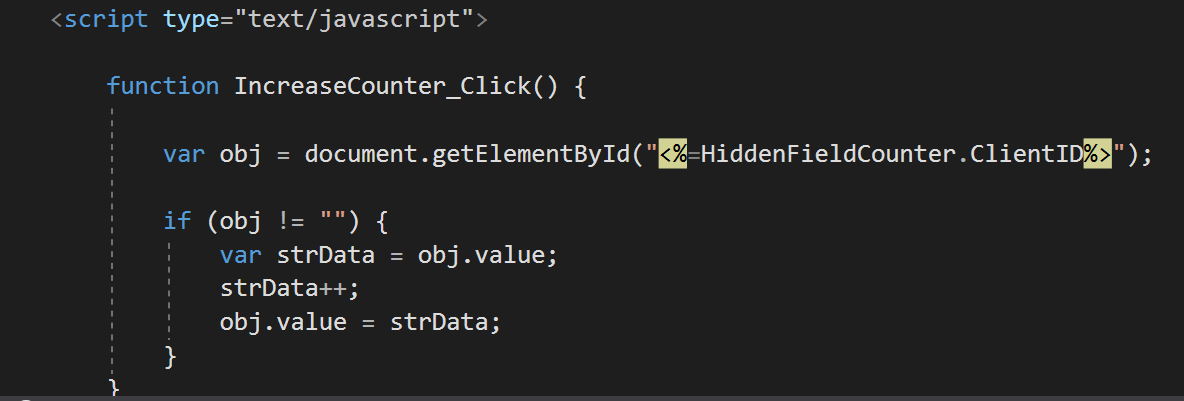


Figure 2: Increment Counter function

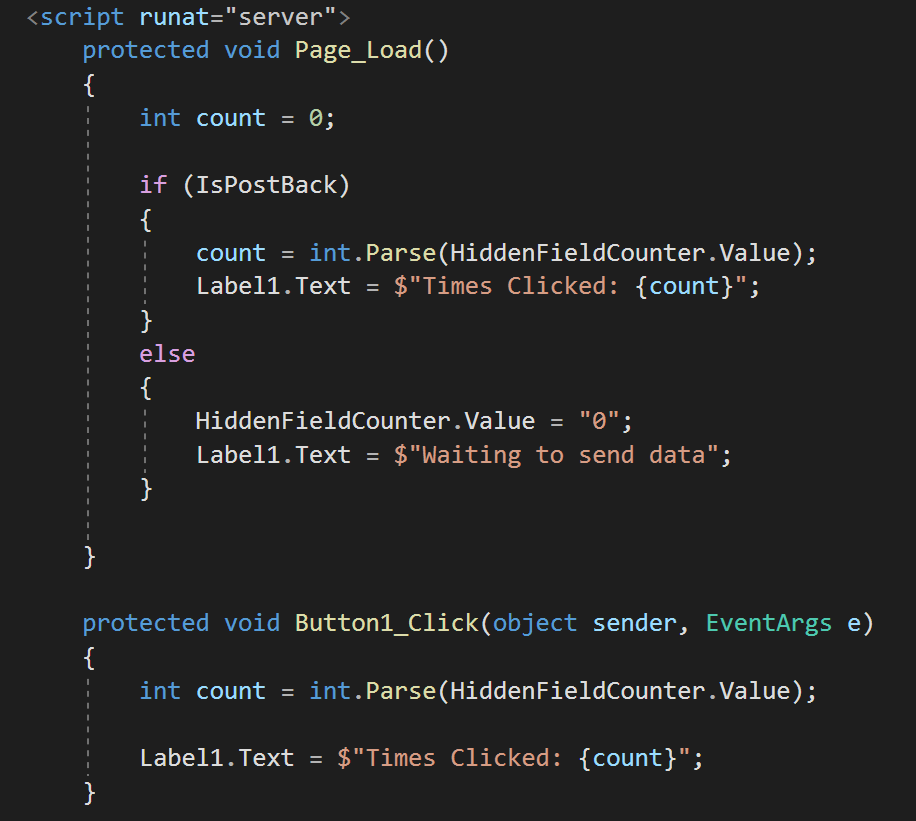


Figure 3: Post back handler and update label with counter

## Session Demonstration

For this demonstration two pages are used to show how information from one page can be saved into a server Session list. Then when another page is loaded, the stored value can be retrieved and used. This can add flexibility for tracking various information across multiple pages without needing the user to keep entering it.

The first form has an input that accepts free input from the user. When the button is clicked the page is posted to the server where it stores the value from the input field into the session list with a key of “FooBar” (Figure 4). The user is then redirected to another page to simulate reading the session value from another page.

When the page loads, it gets the value from the Session using the same key mentioned above and displays the result using a label on the page (Figure 5). The sessions are also cleared to avoid stale state remaining in the session list.

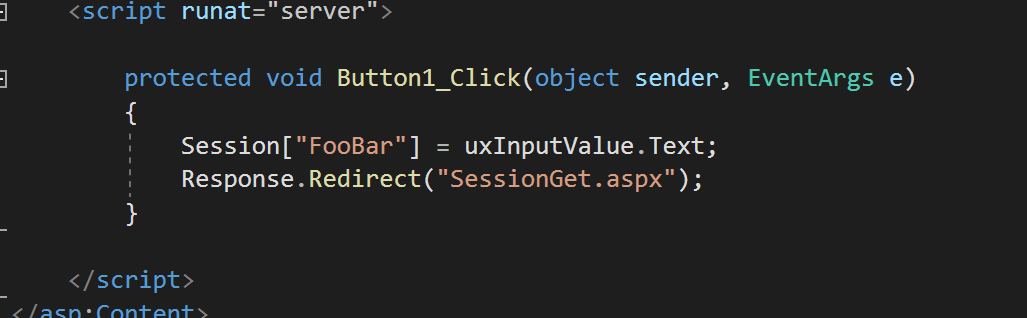


Figure 4: Function to store the value in the session and redirect



Figure 5: Page load used to get session values

## Cookie Demonstration

Cookies offer a method for storing state information in a similar fashion to the Session but instead the information is stored on the client in the browser cache. This is useful for keeping track of the username used to log on to the page.

The format is similar to the Session demo with one page creating a cookie (Figure 6) and then loading another page that reads the cookie data and displays it to the user (Figure 7).

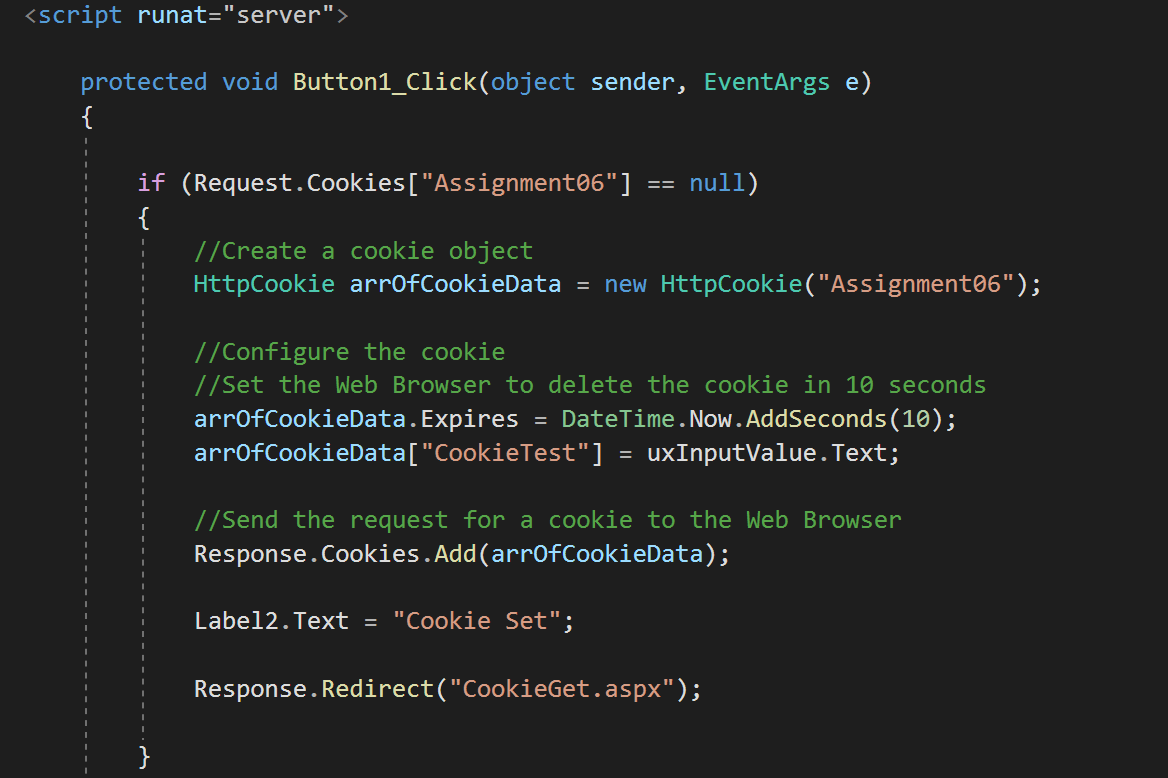


Figure 6: Create a short lived Cookie

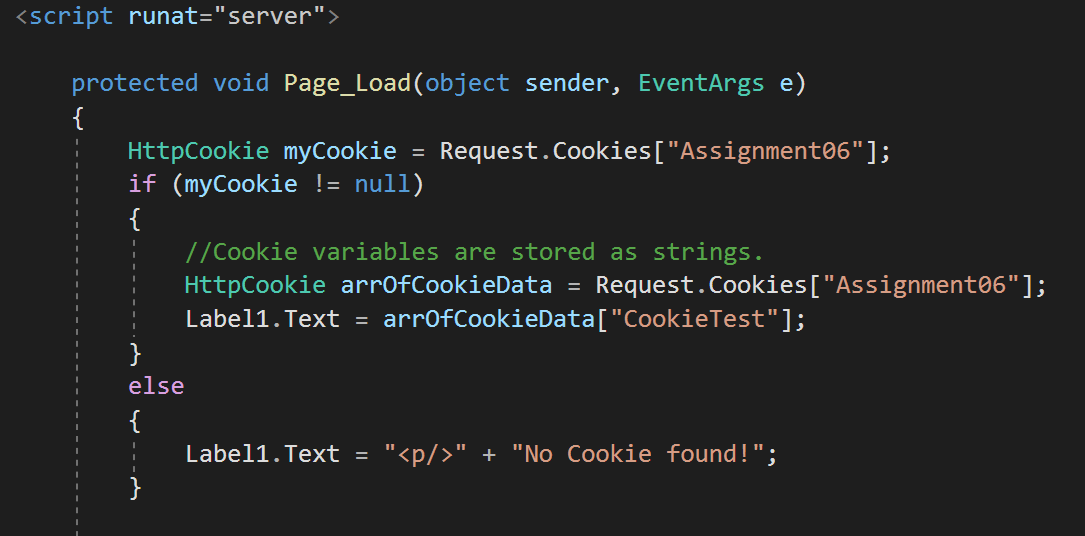


Figure 7: Read cookie data on page load

# Summary

This was just a simple demonstration of each technique that one can build on top of to help track the current application state of the web application to give a more user-friendly experience. Generally, you will make use of multiple techniques to track the state in your web application.