Exercise 08 – The Browser Object Model

Objectives

To use the BOM model to create and read a cookie, combining it with DOM programming to create new DOM elements.

Overview

Informing users you will be holding cookie data about them is a legal requirement within the EU and a commonly required task. In this exercise, we will look at how to check for and access a cookie. Based upon this we will choose whether to conditionally display the cookie info prompt.

This exercise will take around **30 mins**.

Exercise Set Up

1. In VSCode, open the file called **EUCookie.html** from the **Starter** folder inside **Ex08** of the **Exercises** folder.
2. If you have rebooted your computer or stopped *live-server* from running since the last exercise, then use VSCode's integrated terminal to navigate to the **Exercises** folder and run the command **live-server**.
3. Open your browser (if it doesn't automatically) and navigate to **http://localhost:8080** and follow the path to open **Ex08→Starter→ EUCookie.html.**

The workflow for this exercise is as follows:

1. Check for a cookie and if it not present, display some content allowing the user to confirm their understanding;
2. Create a new cookie on agreement.

Part 1 - Check for a cookie

We will start by creating a JavaScript function to check for a cookie.

1. In the script tag in EUCookie.html, create a function called **checkForCookie**.
2. Inside the function body, create a variable **x** and set it to equal **document.cookie**.

The variable **x** will now access the cookie for the current document and return nothing if it is - it is not of much use right now. In the cookie, we are going to add a property to it that will represent the user agreeing to the cookie policy. As any cookie is just a string, we can use string manipulation (covered in Ex02) to check for this content.

3. Under the declaration of x, add an if statement that:
   a) Checks that **x** is *not* an *empty string*;
   b) **AND** checks if **x** *contains* the string **"agreed=true";**
   c) Will execute the body of the loop when both of the above are *not true*.

Your if statement should look like:

```
if (!(x != "" && x.indexOf("agreed=true") !== -1)) {
}
```

If there is no cookie present (i.e. the if statement executes) then we should prompt the user to accept that cookies are used and create the cookie. To do this we will change the hidden property of the appropriate div tag to display the message.

Within the body of the if statement place the following line of code:

```
document.querySelector("#prompt").hidden = false;
```

querySelector is a function of the document that allows us to access particular elements on the page - in this case the element with the id of prompt - more on this in the next chapter!

4. Add a call to the checkForCookie function at the bottom of your code and save it.

Your page should display a green bar at the top with the contents of the div with the id of prompt. Note that clicking the button produces a console error. Why is that?

Part 2 - Create a cookie

The error is generated because we have not defined the function called by the button. On a side note, the button has the type of button as its default value is submit and we don't want that to happen! The function called by button is named **makeCookie** and is called when the *onclick event* of the button is activated (i.e. the button is clicked!) *More on forms and events later in the course.*

1. Under the **checkForCookie** function definition, declare a new function called **makeCookie**.
2. Inside the function body, set **document.cookie** to the string **`agreed=true`**.
3. Change the value of the **hidden** property of the *div with the id* of **prompt** to **true**.
4. Save your file and refer to your browser window - *do not click on the button yet*.
5. Open the *Developer Tools* and click on the **Application** tab.
6. In the list on the left, click on **Cookies** to reveal the list of cookies available (there should be none)
7. Click on the available domain (http://127.0.0.1:8080 - the address of the live-server)

There should be *no cookie data* in the Name or Value column at this point (unless you clicked the button!)

8. You've been itching to, so do it - click the **"Click to confirm"** button!

Seemingly nothing has happened with regards to the cookie, even though the prompt message has gone!

9. Right click anywhere in the *Developer Tools* under the column headings for Cookies and select **Refresh**.

You should see that there is now a **Name** of **agreed** with a **Value** of **true**. Job done!

Well, not quite!

10. Close the browser, reopening and navigating back to the page.

The prompt is back...This is because we have created a temporary cookie and this is removed when the browser is closed. As we probably don't want to have users confirm the same notice every time they visit the site in future, we can add more data to the cookie to ensure that it lives on.

To do this, we will set an expiry date on the cookie using the Date object.

11. The line within **createCookie** where the cookie is created to:

```
document.cookie = `agreed=true;expires=${new Date(2025, 0, 1)}`;
```

12. Save you page and in the browser, delete any previous cookies (right click on the domain on the left in the Developer Tools Application Cookies and select Clear).
13. Refresh the page and see the results - you should see the cookie now has an expiry date.
14. To check the persistence of the cookie, close the browser and reopen the page.
15. If you want to check it again, follow step 12 to remove the cookie.

What happens if you try this in Incognito mode?