Project: Fresh Cookies In The Window: Come And Get ‘Em!

Now that you know all about the browser, the BOM, and the window, let’s get cooking! In this project, you will:

- Set up a server and run files locally
- Practice running a script when the page has loaded
- Set, Get, and Delete cookies using JavaScript
- Open and resize a window using JavaScript

Introduction

Picture yourself baking some fresh cookies (chocolate chip? snickerdoodle? peanut butter?) on a crisp autumn day.

The timer’s gone off, and as you take the piping hot tray out of the oven, you salivate in anticipation of a tasty treat. But, the cookies are way too hot and need to cool down. You crack open the nearest window just enough, and set the tray down next to it, letting a cool breeze waft over the cookies. You’re looking forward to the first bite!

Let the imagery above be your inspiration while completing this coding project. We can think of the browser’s `window`object as a physical window, and browser cookies as actual doughy desserts, if for no other reason that it’s a fun mnemonic device.

Project overview

Let’s practice running a script when the page has loaded. We’ll also manipulate the `window`object by changing its height. Finally, we’ll bake some byte-sized cookies and set them in the window.

Phase 1: Using a local Node/Express server (*Whoa!*)

Express is a Node framework that you'll be using to set up a local server on your machine. This will come in handy when we’re practicing how to set a cookie.

We've set up an Express app for you to use. Open the `cookies-project`folder inside of this Module. Then go through the following:

1. Make sure Node is installed on your machine.
2. Run `npm install`.
3. Run `npm start`, you should see "Example app listening on port 3000!" in your terminal.
4. Open up `localhost:3000`in your browser and make sure you see 'Cookies!!!'

You'll be using the ***public/index.html****file to write your HTML and****public/js/cookies.js****file to write your JavaScript.*

Phase 2: Add a script on DOM/page load (*Practice makes perfect.*)

Inside of your `cookies.js`file:

- Practice running your script after the page has loaded. You can use`DOMContentLoaded`or `window.onload`. If you're linking to an external JS file, practice using the `async`or `defer`methods.

Phase 3: Set a cookie in the window (*Freshly baked!*)

Inside of your `cookies.js`file, practice setting a cookie. Here are a couple of examples:

- `document.cookie = "monster_name=cookie";`
- `document.cookie = "favorite_cookie=snickerdoodle";`

Now looking at your Developer Tools, did the cookies you set appear in the correct place? Try defining one more cookie in your `cookies.js`and refresh your page. Do you see it in the Developer Tools? Now delete your most recent cookie and refresh your page. If you delete a cookie, what happens in the browser tab when you refresh?

Let's show our new cookies to our user! Try using the window.alert()method to let our user know the information of all the cookies stored in the browser.

Phase 4: No such thing as just one more cookie

Setting cookies one at a time by hand is probably getting old by now. Let's write a new function called `setCookie(name, value)`. The `setCookie`function will accept two arguments, a name and a value, and will create a new cookie using those arguments.

Nice! Try testing out your new function by creating a few cookies. Now that we have a way to set cookies it'd sure be nice to have a way to return all the cookies. So... let's hop to it!

Phase 5: Gimme all the cookies

Write a function named `getCookies()`that will return an array of the key value pairs of each set cookie. This is easy to do when you remember that`document.cookie`returns a string!

Look below for an example of how `getCookies`is used:

```
setCookie("dog", "Fido");
setCookie("cat", "Jet");

console.log(getCookies()); // prints ["dog=Fido", "cat=Jet"]
```

Let's take this one step further - what if we wanted to get the value for one particular cookie?

Write a new function `getCookieValue(name)`that intakes that name of a cookie as an argument. If the given cookie name exists `getCookieValue`will return the value of that cookie. If a cookie with the given name doesn't exist the`getCookieValue`function should return `null`.

Here is an example of `getCookieValue`in action:

```
setCookie("cat", "Jet");

```

```
console.log(getCookies()); // prints ["cat=Jet"]
console.log(getCookieValue("cat")); // "Jet"
console.log(getCookieValue("rabbit")); // null
```

Phase 6: Removing a cookie

Now that we have a couple of functions in place to set and get cookies lets write a function called `deleteCookie(name)`. The `deleteCookie`function will accept the name of a cookie to be deleted and will delete that cookie, if it exists. If `deleteCookie`is given the name of a cookie that doesn't exist it should print a message to the user saying the cookie wasn't found.

Here is an example of `deleteCookie`in action:

```
setCookie("cat", "Jet");
setCookie("dog", "Fido");

console.log(getCookies()); // prints ["cat=Jet", "dog=Fido"]
deleteCookie("cat");
console.log(getCookies()); // prints ["dog=Fido"]
deleteCookie("rabbit"); // prints "cookie not found!"
```

We now have set up some nice utility function to get, set, and delete cookies! Pretty yummy if you ask me! 🍪

Bonuses: More fun with cookies!

Check out the MDN documentation on Document.cookieto help you complete the tasks below.

Bonus A: Open a new window and resize it based on cookie value (*Smells great!*)

Let's trying doing something fun will all our new cookie functions. Let's write some code to do the following:

1. generate a random number
2. store that number in a cookie
3. open a new window and set the new window's height and width to the cookie value we stored in step 2.

- *Hint: Recall that we can use Math.random()to generate a random number.*
- *Hint: Use* `window.open()`*AND* `window.resizeTo()`*OR* `window.resizeBy()`

When you run the script, you should see a new window open with the width and height set to the value that was stored in the cookie. Baked and cooled!

Bonus B: Create new cookies at the click of a button

Are you a master baker? Create the necessary html elements and JavaScript so that a user can input a cookie name and value and click a button to create a cookie with the name and value they specified.

The sequence of events will be:

1. A user inputs the name and value they want for their cookie
2. The user clicks the button to create their new cookie
3. The new cookie is created

4. To ensure the best user experience, empty the two inputs for the cookie name and value (so a user can easily create another cookie)
5. Log all the cookies to the console so the user can see their new cookie!

Congratulations! Hang up your your apron for now knowing you are a cookie master.