Slim

How to Manage Cookies in JavaScript?

Quick & Easy Guide for Beginners!

Slim toumi
Slim toumi

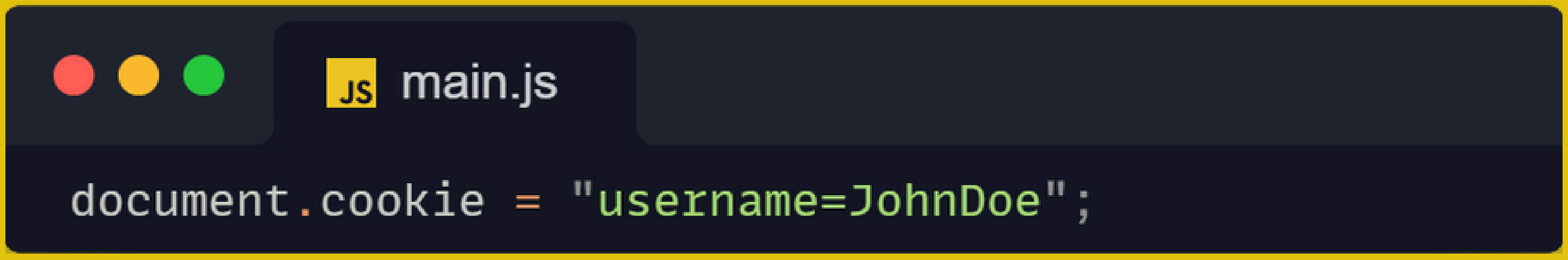
What is cookies ?

Cookies are small pieces of data stored in a user's browser to remember information, like login status or preferences, between visits. Here’s a guide to working with cookies in JavaScript!

Why Use cookies?

- **Persist data:** Keep data like user preferences or session info across page loads
- **Track activity:** Understand user behavior to enhance experience.
- **Remember sessions:** Enable users to stay logged in.

Creating cookies



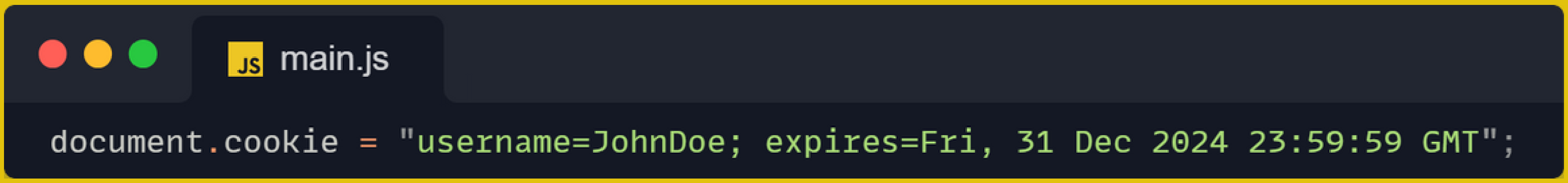
```
document.cookie = "username=JohnDoe";
```

Note:
- Sets a cookie with a key of username and a value of JohnDoe.
- Default expires when the session ends (when the browser is closed).

Setting a Cookie with Expiration

Cookies can have expiration times using either the **expires** attribute (a fixed date) or **max-age** (duration in seconds).

1. expires: Sets a specific expiration date for the cookie.

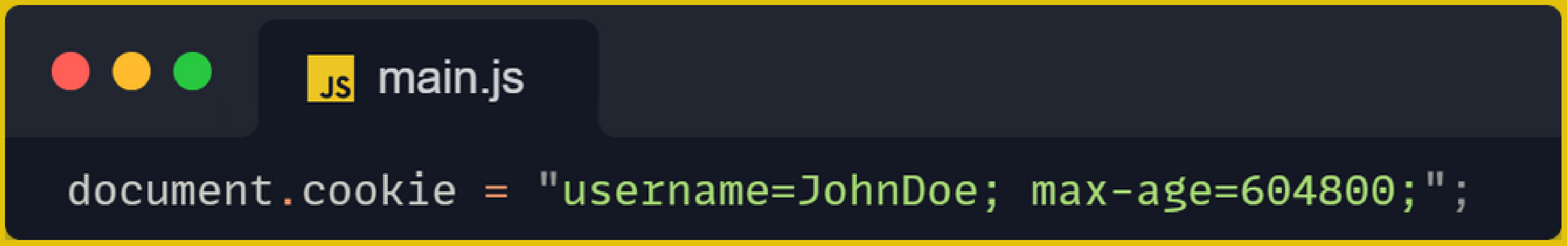


```
document.cookie = "username=JohnDoe; expires=Fri, 31 Dec 2024 23:59:59 GMT";
```

Note:

- The expires attribute takes a date in UTC string format.

2. max-age: Sets the expiration time relative to when it's set (in seconds).



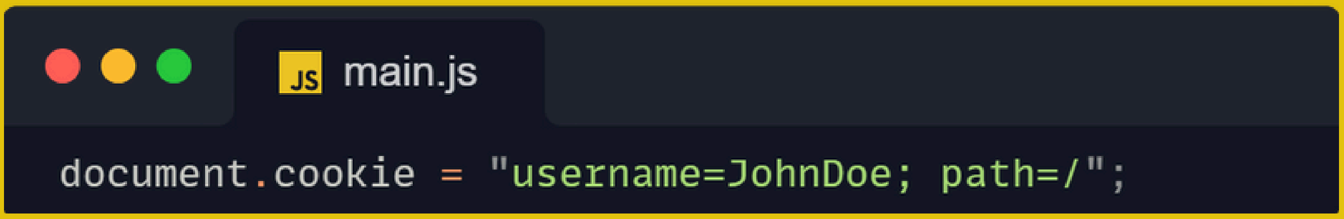
```
document.cookie = "username=JohnDoe; max-age=604800;";
```

Note:

- Expires in 7 days (604800 seconds)
- If neither **expires** nor **max-age** is set, the cookie will be a session cookie and will be deleted when the browser is closed.

Attributes for Cookies

1. path: Defines the scope of the cookie (i.e., the paths where it's accessible)..



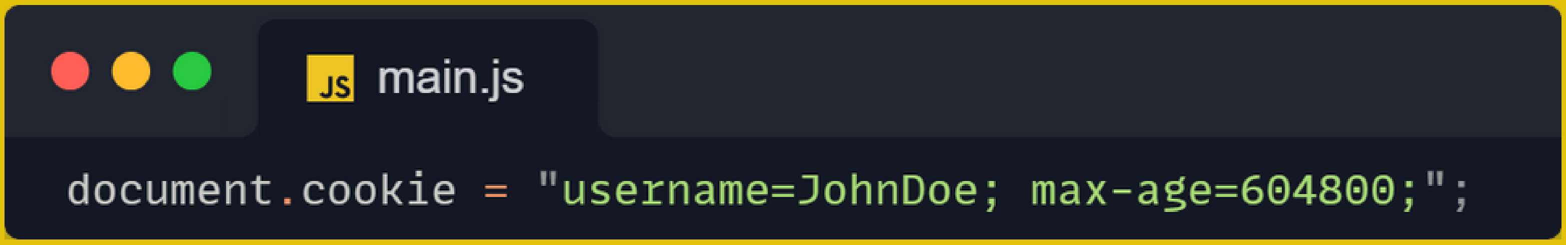
main.js

```
document.cookie = "username=JohnDoe; path=/";
```

Note:

- **path=/**: Makes the cookie accessible on all pages of the website.

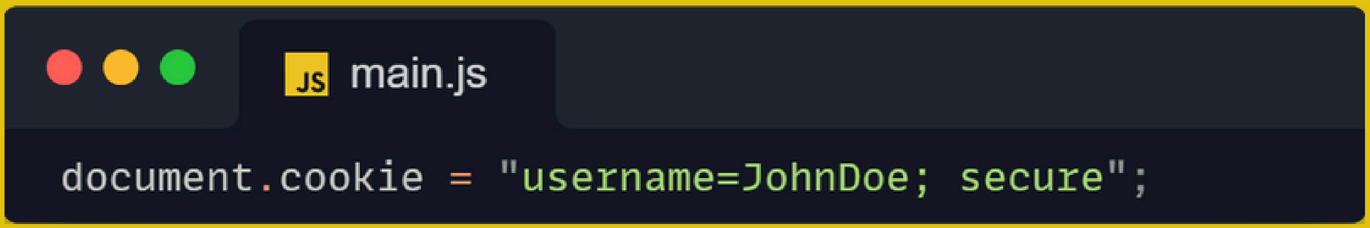
2. domain: Specifies the domain for which the cookie is valid. This helps control cross-subdomain access.



main.js

```
document.cookie = "username=JohnDoe; max-age=604800;";
```

3. secure: Ensures the cookie is only sent over HTTPS.



main.js

```
document.cookie = "username=JohnDoe; secure";
```

4. httpOnly: Makes the cookie inaccessible to JavaScript (document.cookie), enhancing security.

Note:

- This attribute is set from the server side, not in JavaScript.

Slim

Reading Cookies

To read cookies, access document.cookie, which returns a string of all cookies in key-value pairs.

main.js

```javascript
const allCookies = document.cookie;
//example: Returns "username=JohnDoe; theme=dark"
```

Parsing Cookies: To get a specific cookie value, split document.cookie as follows:

main.js

```javascript
function getCookie(name) {
  const cookies = document.cookie.split("; ");
  for (let cookie of cookies) {
    const [key, value] = cookie.split("=");
    if (key === name) return value;
  }
  return null;
}
```

Usage:

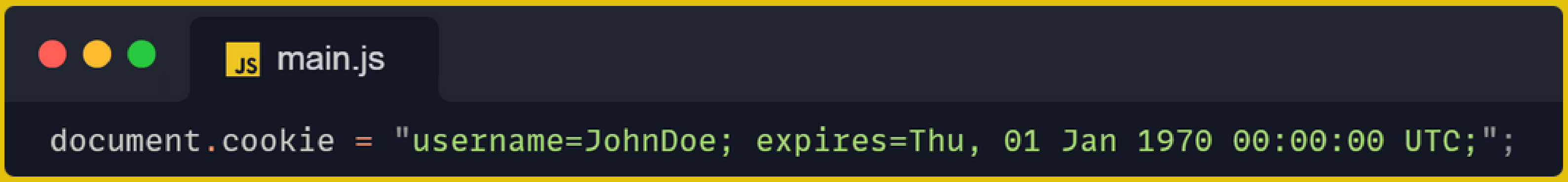
main.js

```javascript
const username = getCookie("username"); // Returns "JohnDoe"
```

Slim toumi
Slim toumi

Deleting Cookies

To delete a cookie, set its expiration date to a past date.



main.js

```
document.cookie = "username=JohnDoe; expires=Thu, 01 Jan 1970 00:00:00 UTC;";
```

Best Practices

- **Limit usage**: Keep cookies small to reduce load time.
- **Secure data**: Only store non-sensitive data, and use HTTPOnly and Secure attributes when possible.
- **Use alternatives when possible**: For complex data, consider localStorage or sessionStorage.

Hopefully You Found It Usefull!

Be sure to save this post so you can come back to it later

like Comment Share

Slim toumi
Slim toumi