Understanding Cookies in Web Development

Software Development Bootcamp

An Overview of Uses, Limitations, and Implementation

Topic

Cookies

What Are Cookies?

- Small pieces of data stored as text files on the client-side (user's browser)
- Created by web servers and sent to the browser with the server's response
- Sent back to the server with every subsequent request to the same domain
- Typically contain:
    - Name-value pairs of information
    - Expiration date
    - Domain and path of the server it should be sent to

Documentation

Key Characteristics Of Cookies

- **Size**: Usually limited to 4KB
- **Persistence**: Can be session cookies (temporary) or persistent cookies (with expiration date)
- **Scope**: Bound to specific domains and paths

How Cookies Work

- Server sends a response with a Set-Cookie header
- Browser stores the cookie
- Browser sends the cookie back in the Cookie header of future requests
- Server can read, update, or delete the cookie

Cookie Uses

- Tracking users across time.
- Storing data without using server space.
- An early version of LocalStorage for client-side storage.

Cookie Limitations: Security

Not inherently secure; can be vulnerable to various attacks:

- **Cross-Site Scripting (XSS)**: Malicious scripts can access cookies if not properly secured
- **Cross-Site Request Forgery (CSRF)**: Unauthorized actions can be performed using the user's cookies
- **Man-in-the-Middle attacks**: Cookies can be intercepted if not transmitted over HTTPS

Mitigation:

- Use the HttpOnly flag to prevent JavaScript access to cookies
- Implement the Secure flag to ensure cookies are only transmitted over HTTPS
- Utilize SameSite attribute to protect against CSRF attacks

Cookie Limitations: Size

- Typically limited to 4KB (4096 bytes) per cookie
- Browsers usually limit the number of cookies per domain (often around 50)

Implications:

- Restricts the amount of data that can be stored
- May require splitting data across multiple cookies or using alternative storage methods for larger datasets

Cookie Limitations: Privacy And Compliance

- Subject to privacy laws and regulations (e.g., GDPR, CCPA)
- Require user consent in many jurisdictions
- Users can disable or delete cookies, potentially disrupting site functionality

Cookie Limitations: Cross-Origin Restrictions

- Same-Origin Policy limits access to cookies across different domains
- Third-party cookies (set by a domain other than the one being visited) are increasingly blocked by browsers and ad-blockers

Implications:

- Challenges for tracking and advertising technologies
- Difficulties in implementing single sign-on across multiple domains

Cookie Sessions

- Client-side cookie data is insecure, limited, and unstable.
- Often used as a key to a server-side database (session cookie).
- Implements 'Remember Me' login checkboxes.

Other Cookie Limitations

Performance:

- Large or numerous cookies can increase network traffic and slow down web applications

Limited Persistence:

- Browsers may clear cookies periodically or upon closing
- Users can manually delete cookies at any time
- Expiration dates can be ignored or overwritten by the browser or user

Lack Of Real Time Synchronization:

- Changes to cookies on the server are not immediately reflected on the client
- Requires a page refresh or new request to update cookie data

Cookie Headers

Request Header:

- Includes all the cookies the browser has saved for that website
- Example: `Cookie: ice_cream=chocolate; fruit=banana`

Response Header:

- Tells the browser to save a new cookie or update an existing one
- Example: `Set-Cookie: ice_cream=chocolate`

Fetch API & Cookies

- Fetch API does not always send cookies due to same-origin policy.
- Use credentials: include to send cookies:
- **WARNING**: Only use `credentials: include` if necessary and never in production without a good reason.

```
// Fetch request using 'credentials: include'
fetch('/articles.json', {
 credentials: 'include'
})
```

Express & Cookies

- The request's Cookie header is available via request.headers.cookie
- Use cookie-parser middleware to parse cookies
- Turns complicated cookie data into an easy to use format

```
const express = require('express');
// Requiring cookieParser
const cookieParser =
require('cookie-parser');
const app = express();
// Using cookieParser middleware
app.use(cookieParser());
app.get('/', function(request, response) {
 console.log('Cookies: ', request.cookies);
});
```

Setting Cookies on the Server

- Use response.cookie method:
- Express handles JSON encoding and URI encoding.

```
app.get('/', function(request, response) {
 console.log('Cookies: ', request.cookies);
 response.cookie('cart', ['milk',
'carrots', 'dog food'])
});
```

Cookie Counter

- Detailed example: cookie_counter.js is a simple hit counter.
- Run the example:
  - git clone Cookie Counter Repo
  - cd cookie-counter
  - npm install
  - npm start
- Visit http://localhost:5000/cookie.html and reload the page to see it in action.

Exercise

Creating Cookies