Cookies і Web Storage

№ уроку:	14	**Курс:**	JavaScript Базовий

Засоби навчання:	Visual Studio Code Web Browser

Огляд, мета та призначення уроку

Навчитися зберігати дані в браузері за допомогою cookie та We Storage API.

Вивчивши матеріал цього заняття, учень зможе:

- Розуміти, що таке cookie.
- Використовувати методи encodeURIComponent і decodeURIComponent.
- Працювати з cookie сторінки через JavaScript-код.
- Використовувати localStorage та sessionStorage для зберігання даних.
- Серіалізувати та десеріалізувати дані у JSON-формат.

Зміст уроку

1. Що таке cookie.
2. Робота з cookie через JavaScript-код.
3. Огляд Web Storage API.
4. sessionStorage та localStorage.

Резюме

- **Cookie** – невеликий фрагмент даних, який надсилається веб-сервером і зберігається на стороні клієнта. Веббраузер відправляє cookie щоразу, коли виконує запит на сайт, з якого було отримано cookie.

- Доступ до cookie через JavaScript можна отримати за допомогою властивості **document.cookie.**

- Найчастіше cookie використовуються для **авторизації користувача**. Коли користувач заходить на закриту частину сайту, сервер перевіряє наявність спеціального значення, яке зберігається в cookie. Якщо цього значення немає, користувача перенаправляють на сторінку входу на сайт. Якщо значення є, показують вміст закритої частини сайту.
  Якщо користувач, потрапивши на сторінку входу, вводить правильний логін та пароль (користувач з таким логіном та паролем є в базі), тоді сервер повертає користувачеві cookie авторизації, які зберігаються в пам'яті браузера. За кожного наступного запиту на сервер цей cookie буде частиною запиту. Сервер розумітиме, що користувач виконав вхід на сайт і що йому можна показувати закриті сторінки.

- **Cookie** передається на сервер як **HTTP-заголовок**.

- Значення одного cookie не має перевищувати 4 Кб. Більшість браузерів дозволяють створювати орієнтовно 20 cookie на один домен.

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine

Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- Cookie не призначені для зберігання великих даних. Вони більше підходять для відстеження ідентифікатора користувача під час переходів за різними сторінками сайту або збереження незначних блоків даних.

- Cookie представляється рядком, який складається з ключів і значень, розділених крапкою з комою.

- **Атрибути cookie:**
  - **path** – вказує сторінки, на яких буде працювати cookie. catalog на сторінках /catalog/folder тощо, але на сторінці home не працюватиме для всіх сторінок;
  - **max-age** – час життя значення в секундах;
  - **expires** – дата, коли значення має бути видалено. Приклад значення: 20 May 2022 5:15:05 GMT. Для видалення cookie необхідно встановити минулу дату або max-age=0;
  - **samesite** – значення strict, lax чи none. Налаштування необхідне для захисту від уразливості CSRF. Атрибут не підтримується старими браузерами (до 2017 року);
  - **secure** – cookie можна надсилати лише за HTTPS;
  - **httpOnly** – cookie використовуються тільки для HTTP-запитів і не можуть бути отримані за допомогою JS-коду.

- **Приклад створення cookie з параметрами:**
  document.cookie = "color=green; path=/; max-age=30;" – назва color, значення – green, відправлятиметься на сервер під час запитів до будь-якої сторінки та видаляється з пам'яті через 30 секунд після створення.

- Cookie можуть містити лише текстові значення. Для правильного подальшого читання значення мають бути закодовані за допомогою методу **encodeURIComponent**. Під час читання раніше закодованого значення використовується метод **decodeURIComponent**. **encodeURIComponent** – метод кодування рядка в компонент, який буде використовуватися в URI. Цей метод замінює всі символи, окрім символів латинського алфавіту, цифр і символів _ . ! ~ * ' ( ).

  Метод використовується для запобігання некоректних запитів, наприклад:
  - рядок без використання URI кодування – https://itvdn.com/search?term=position=absolute;
  - рядок без використання URI кодування – https://itvdn.com/search?term=position%3Dabsolute.

  У другому рядку символ = був замінений на %3D, оскільки цей символ використовується для визначення значення для ключа. У певних ситуаціях це може вплинути на правильне розпізнавання ключів та їхніх значень.

- **Web Storage API** – механізм, який дає змогу створювати пари ключ-значення та зберігати їх у пам'яті браузера, використовуючи інтуїтивно зрозуміліший синтаксис, ніж під час роботи з cookies.

- **Є два механізми зберігання даних на клієнті:**

1. **sessionStorage**. Створює окреме сховище для кожного джерела, яке наявне, поки наявна сторінка (під час перевантаження сторінки інформація не видаляється). Дані sessionStorage видаляються під час закриття браузера. Дані sessionStorage не передаються на сервер.

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine
Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Обмеження сховища до 5 Мб, але може відрізнятися, залежно від браузера.

2. **localStorage**. Те ж саме, що і sessionStorage, але інформація зберігається в сховищі безстроково та залишається доступною після перезапуску браузера. Дані видаляються зі сховища за допомогою JavaScript або під час чищення кеша браузера.

- **Origin (джерело)** – джерело, з якого здійснюється завантаження контенту. Містить протокол, ім'я хоста, порт.
  Наприклад, http://localhost:88, https://itvdn.com, http://example.com, https://example.com. Усі ці значення належать до різних джерел. Браузер створює сховище окремо для кожного джерела за умови, що з джерела було отримано код, який звернувся до вебсховища.

- **Методи роботи з localStorage і sessionStorage:**
  - **setItem(key, value)** – зберігає значення за ключем;
  - **getItem(key)** – повертає значення за ключем або null, якщо значення немає;
  - **removeItem(key)** – видаляє запис за ключем;
  - **clear()** – видаляє всі записи для поточного джерела;
  - **key(index)** – повертає ім'я ключа за вказаним індексом;
  - **length** – повертає кількість ключів у сховище.

- **JSON (JavaScript Object Notation)** – формат зберігання даних, який ґрунтується на JavaScript. Застосовується для зберігання та передання даних між сервером і клієнтом. Не залежить від мови програмування.
  JSON.stringify(object) може використовуватися для перетворення об'єкта або масиву в формат JSON. Для перетворення значення з формату JSON на об'єкт використовується метод JSON.parse(json_string).
  Оскільки вебсховище може працювати лише з рядковими значеннями, під час збереження об'єктів може знадобитися використання цих методів.

- Отримати доступ до cookies та web storage можна за допомогою інструментів розробника у вкладці Application.

Закріплення матеріалу

- Що таке cookie? Як вони працюють?
- Наведіть приклад завдання, для якого можуть використовуватися cookies.
- Як можна видалити cookie за допомогою коду JavaScript?
- Для чого потрібен метод encodeURIComponent?
- Скільки часу зберігаються дані в sessionStorage, а скільки в localStorage?
- Які ліміти на обсяг збереженої інформації наявні для вебсховищ?
- Що таке джерело?
- Що таке JSON? Як зберегти та відновити об'єкт, використовуючи цей формат у JavaScript?

Самостійна діяльність учня

Виконайте завдання у директорії Tasks\017 Cookies and Web Storage. Текст завдань розташований у коментарях, у тегах script.

Рекомендовані ресурси

CSRF
https://ru.wikipedia.org/wiki/Межсайтовая_подделка_запроса

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine
Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Властивість samesite
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie/SameSite

Cookie
https://developer.mozilla.org/ru/docs/Web/API/Document/cookie

Web Storage API
https://developer.mozilla.org/ru/docs/Web/API/Web_Storage_API

Origin (джерело)
https://developer.mozilla.org/ru/docs/Web/HTTP/Headers/Origin

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine

Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com