WEB STORAGE API

The Web Storage API is a web technology that provides a simple key-value storage mechanism for web applications. It allows developers to store data on the client-side (in the user's browser) to maintain state and data persistence across different sessions.

The Web Storage API consists of two main mechanisms:

Local Storage

The Local Storage mechanism allows you to store data on the client's device without an expiration date. The data stored in Local Storage remains even after the browser is closed and is accessible across tabs and windows of the same origin. It has a larger storage capacity compared to cookies, typically around 5-10 MB, depending on the browser. To use Local Storage, you can use the ***localStorage object in JavaScript.***

Local Storage	
Capacity	10mb
Browsers	HTML5
Accessible from	Any window/tab
Expires	Never
Storage Location	Brower
Send with	No

Local Storage Methods:

- **localStorage.setItem(key, value):** This method allows you to store a key-value pair in the Local Storage. The key is a string that serves as the identifier for the data, and the value can be a string or an object that you want to store.
  e.g.
  // Storing a value with a key in localStorage
  localStorage.setItem('username', 'JohnDoe');

- **localStorage.getItem(key):** Retrieves the value associated with the given key from Local Storage. If the key does not exist, it returns null.
  e.g.
  // Retrieving a value from localStorage using its key
  var username = localStorage.getItem('username');
  console.log(username); // This will output the value stored with the key 'username'

- **localStorage.removeItem(key):** Removes the key-value pair associated with the given key from Local Storage.
  e.g.
  // Removing an item from localStorage using its key
  localStorage.removeItem('username');

- **localStorage.clear():** Clears all data stored in the Local Storage, effectively resetting it.
  e.g.
  // Clearing all data stored in localStorage
  localStorage.clear();

It's important to note that localStorage is persistent storage, meaning the data will remain even after the user closes the browser window or navigates away from the page.

This makes it useful for storing things like user preferences, authentication tokens, or any other data that you want to persist across sessions.

However, because it persists across sessions and domains**, it's important to be mindful of what data you store and ensure that sensitive information is properly secured.**

Session Storage:

The Session Storage mechanism allows you to store data on the client's device for the **duration of a single session.**

A session lasts as long as the browser is open and the user is browsing the website. Once the browser is closed, the data stored in Session Storage is cleared. Like Local Storage, it is also accessible across tabs and windows of the same origin.

Session Storage	
Capacity	5mb
Browsers	HTML5
Accessible from	Same window/tab
Expires	Close on same window/tab
Storage Location	Brower
Send with request	No

To use Session Storage, you can use the sessionStorage object in JavaScript. Both Local Storage and Session Storage are part of the Web Storage API and provide a simple way for web applications to store and retrieve data on the client-side. They are more efficient and flexible than using cookies for storing larger amounts of data and are commonly used for tasks like caching, saving user preferences, and maintaining application state.

Session Storage Methods:

- sessionStorage.setItem(key, value): This method allows you to store a key-value pair in the Session Storage. It works similarly to localStorage.setItem(), but the data will only be available for the duration of the session.
  e.g.
  // Storing a value with a key in sessionStorage
  sessionStorage.setItem('username', 'JohnDoe');

- **sessionStorage.getItem(key):** Retrieves the value associated with the given key from the Session Storage. If the key does not exist, it returns null.
  e.g.
  // Retrieving a value from sessionStorage using its key
  var username = sessionStorage.getItem('username');
  console.log(username); // This will output the value stored with the key 'username'

- **sessionStorage.removeItem(key):** Removes the key-value pair associated with the given key from the Session Storage.
  e.g.
  // Removing an item from sessionStorage using its key
  sessionStorage.removeItem('username');

- **sessionStorage.clear():** Clears all data stored in the Session Storage, effectively resetting it.
  e.g.
  // Clearing all data stored in sessionStorage
  sessionStorage.clear();

Menu based Program to demonstrate of local and session storage

e.g.

```
<!DOCTYPE html>
<html>
<head>
  <title>Storage Menu</title>
</head>
<body>
  <h1>Storage Menu</h1>
  <h2>Session Storage</h2>
  <ul>
    <li><button onclick="setSessionStorage()">Set Session Storage</button></li>
    <li><button onclick="readSessionStorage()">Read Session Storage</button></li>
    <li><button onclick="modifySessionStorage()">Modify Session Storage</button></li>
    <li><button onclick="deleteSessionStorage()">Delete Session Storage</button></li>
  </ul>
  <p id="sessionStorageValue"></p>

  <h2>Local Storage</h2>
  <ul>
    <li><button onclick="setLocalStorage()">Set Local Storage</button></li>
    <li><button onclick="readLocalStorage()">Read Local Storage</button></li>
    <li><button onclick="modifyLocalStorage()">Modify Local Storage</button></li>
    <li><button onclick="deleteLocalStorage()">Delete Local Storage</button></li>
  </ul>
  <p id="localStorageValue"></p>

  <script>
    // Session Storage
    function setSessionStorage() {
      const key = prompt("Enter session storage key:");
      const value = prompt("Enter session storage value:");

      sessionStorage.setItem(key, value);
      alert(`Item "${key}" set in session storage with value "${value}".`);
    }

    function readSessionStorage() {
      const key = prompt("Enter session storage key to read:");
      const value = sessionStorage.getItem(key);

      if (value !== null) {
        document.getElementById("sessionStorageValue").textContent = `Value of "${key}" in
session storage: ${value}`;
      } else {
        document.getElementById("sessionStorageValue").textContent = `Item "${key}" not found
in session storage.`;
      }
    }
```

```
    function modifySessionStorage() {
      const key = prompt("Enter session storage key to modify:");
      const newValue = prompt(`Enter new value for "${key}" in session storage:`);

      sessionStorage.setItem(key, newValue);
      alert(`Value of "${key}" in session storage modified to "${newValue}".`);
    }

    function deleteSessionStorage() {
      const key = prompt("Enter session storage key to delete:");

      sessionStorage.removeItem(key);
      alert(`Item "${key}" deleted from session storage.`);
    }

    // Local Storage
    function setLocalStorage() {
      const key = prompt("Enter local storage key:");
      const value = prompt("Enter local storage value:");

      localStorage.setItem(key, value);
      alert(`Item "${key}" set in local storage with value "${value}".`);
    }

    function readLocalStorage() {
      const key = prompt("Enter local storage key to read:");
      const value = localStorage.getItem(key);

      if (value !== null) {
        document.getElementById("localStorageValue").textContent = `Value of "${key}" in local
storage: ${value}`;
      } else {
        document.getElementById("localStorageValue").textContent = `Item "${key}" not found in
local storage.`;
      }
    }

    function modifyLocalStorage() {
      const key = prompt("Enter local storage key to modify:");
      const newValue = prompt(`Enter new value for "${key}" in local storage:`);

      localStorage.setItem(key, newValue);
      alert(`Value of "${key}" in local storage modified to "${newValue}".`);
    }

    function deleteLocalStorage() {
      const key = prompt("Enter local storage key to delete:");

      localStorage.removeItem(key);
      alert(`Item "${key}" deleted from local storage.`);
```

```
    }
  </script>
</body>
</html>
```

Akhilesh Kumar Gupta
(Technical Training specialist)
TechCarrel LLP