



Almacenamiento Web

Centro de Servicios y Gestión Empresarial
SENA Regional Antioquia



www.sena.edu.co

Contextualización

Almacenamiento Web

El almacenamiento web permite a las aplicaciones guardar información localmente en el navegador del usuario, sin necesidad de un servidor. Esto mejora la experiencia del usuario al conservar datos entre sesiones o durante una navegación activa.



Almacenamiento Web



Los navegadores modernos ofrecen diferentes formas de almacenamiento local:

localStorage: guarda datos de forma persistente.

sessionStorage: guarda datos temporales por pestaña.

cookies: pequeñas piezas de datos que pueden ser enviadas al servidor.

indexedDB: base de datos local para grandes volúmenes.

Cache API: almacena archivos para apps offline.

Almacenamiento Web



¿Qué necesitas?	Solución recomendada
Guardar preferencia de usuario (tema oscuro, idioma)	localStorage
Guardar datos temporales mientras el usuario navega	sessionStorage
Autenticación y login	Cookies
Almacenar historial, archivos, datos complejos	IndexedDB
Crear una app que funcione offline	Cache API + IndexedDB

Almacenamiento Web



Almacenamiento	Tamaño	Persistencia	Ideal para
localStorage	~5-10 MB	Permanente	Configuración, preferencias, listas
sessionStorage	~5 MB	Solo sesión/tab	Formularios, paso a paso, navegación temporal
Cookies	~4 KB por cookie	Configurable	Autenticación, sesión, tracking
IndexedDB	Cientos de MB	Permanente	Grandes volúmenes de datos, apps offline
Cache API	Variable	Permanente	Archivos estáticos offline (PWA)

Objects ({ })

Los objetos permiten almacenar datos como pares clave-valor, facilitando la representación de información estructurada.

```
let persona = {  
    nombre: "Carlos",  
    edad: 30,  
    ciudad: "Medellín"  
};  
console.log(persona.nombre); // "Carlos"  
console.log(persona["edad"]); // 30  
// Agregar, Modificar y Eliminar Propiedades  
persona.apellido = "Gómez";  
persona.edad = 31;  
delete persona.ciudad;  
console.log(persona);
```

Local Storage

LocalStorage

localStorage es una API del navegador que te permite almacenar pares clave-valor en el navegador del usuario de forma permanente (hasta que se elimine manualmente o por código).

- Los datos no se borran al cerrar el navegador.
- Solo se pueden guardar strings (por lo tanto, hay que convertir objetos o arrays con *JSON.stringify()*)
- *JSON.stringify()* es un método que convierte un objeto o valor en una cadena JSON (JavaScript Object Notation).

Operaciones Básicas de localStorage

Acción	Código	Descripción
Guardar dato	<code>localStorage.setItem("clave", "valor")</code>	Guarda un dato
Leer dato	<code>localStorage.getItem("clave")</code>	Obtiene un dato
Eliminar uno	<code>localStorage.removeItem("clave")</code>	Borra una clave específica
Eliminar todo	<code>localStorage.clear()</code>	Borra todo el almacenamiento
Comprobar existencia	<code>localStorage.getItem("clave") != null</code>	Verifica si existe un dato

Ejemplo de localStorage

```
let persona = {  
    nombre: "Juan",  
    edad: 30,  
    profesion: "Programador"  
};  
  
// Guardar en LocalStorage  
localStorage.setItem("persona", JSON.stringify(persona));  
  
// Recuperar de LocalStorage  
let personaRecuperada = JSON.parse(localStorage.getItem("persona"));  
console.log(personaRecuperada);  
// {nombre: "Juan", edad: 30, profesion: "Programador"}  
  
// Eliminar de LocalStorage  
localStorage.removeItem("persona");  
  
// Limpiar LocalStorage  
localStorage.clear();
```

¿Dónde se ve el Local Storage en el navegador?

1. Abrir el navegador (Chrome recomendado).
2. Presionar F12 (abrir las herramientas del desarrollador).
3. Ir a la pestaña "Aplicación" o "Application".
4. En el panel izquierdo, hacer clic en "Local Storage" > [sitio_web].
5. Ver las claves y valores guardados.

Ejercicio

Agenda de Contactos con Local Storage

Desarrollar una aplicación que permita gestionar la agenda de contactos de un usuario; Las opciones disponibles de la agenda son:

- Agregar un nuevo contacto
- Mostrar la agenda de contactos
- Eliminar la agenda de contactos

LocalStorage

¿Qué se puede hacer con Local Storage?

- Guardar listas de tareas, carrito de compras, configuraciones, temas oscuros, historial de uso, etc.
- Evitar usar base de datos o backend para prototipos rápidos.
- Hacer que una app web recuerde datos entre sesiones.

Session Storage

Session Storage

sessionStorage es una API del navegador similar a `localStorage`, que permite guardar datos temporales (pares clave-valor) en el navegador.

- Los datos de *sessionStorage* solo se conservan durante la sesión actual del navegador.
- Si el usuario cierra la pestaña o ventana, los datos se eliminan automáticamente.

Comparación localStorage y sessionStorage

Característica	localStorage	sessionStorage
Duración	Persistente (hasta que se borre)	Solo durante la sesión/tab actual
Alcance	Toda la aplicación/sitio	Solo en la pestaña actual
Tamaño aproximado	5-10 MB	5 MB
Uso típico	Datos persistentes: usuarios, temas, historial	Datos temporales: formularios, pasos de navegación
Accesible desde otra pestaña	<input checked="" type="checkbox"/> Sí	✗ No

Operaciones Básicas de sessionStorage

Acción	Código	Descripción
Guardar un dato	<code>sessionStorage.setItem("clave", "valor")</code>	Guarda un valor en la sesión actual
Obtener un dato	<code>sessionStorage.getItem("clave")</code>	Recupera el valor asociado a la clave
Eliminar un dato	<code>sessionStorage.removeItem("clave")</code>	Elimina la clave y su valor correspondiente
Eliminar todo	<code>sessionStorage.clear()</code>	Borra todos los datos de la sesión actual
Verificar existencia	<code>sessionStorage.getItem("clave") !== null</code>	Verifica si existe una clave
Guardar array/objeto	<code>JSON.stringify()</code>	Convierte estructuras a string para almacenarlas
Leer array/objeto	<code>JSON.parse(sessionStorage.getItem("clave"))</code>	Convierte string de vuelta a estructura original

Ejemplo de sessionStorage

```
let persona = {  
    nombre: "Juan",  
    edad: 30,  
    profesion: "Programador",  
};  
  
// Guardar en SessionStorage  
let datos = JSON.stringify(persona);  
sessionStorage.setItem("usuario", datos);  
  
// Recuperar de SessionStorage  
let personaRecuperada = JSON.parse(sessionStorage.getItem("usuario"));  
console.log(personaRecuperada);  
  
// Eliminar de SessionStorage  
sessionStorage.removeItem("usuario");  
  
// Limpiar SessionStorage  
sessionStorage.clear();
```

Ejercicio

Formulario de Registro Multipaso con sessionStorage

Simular un formulario dividido en varios pasos (por ejemplo, como los de matrícula, reservas o inscripción). Necesitas guardar temporalmente los datos que el usuario va ingresando, para que si cambia de paso, no se pierda la información. Los datos deben eliminarse cuando el usuario cierra la pestaña o recarga.

Paso 01: Nombre y correo electrónico

Paso 02: Ciudad y teléfono



G R A C I A S

Presentó: Alvaro Pérez Niño

Instructor Técnico

Correo: aperezn@sena.edu.co

<http://centrodesserviciosygestionempresarial.blogspot.com/>

Línea de atención al ciudadano: 01 8000 910270

Línea de atención al empresario: 01 8000 910682



www.sena.edu.co