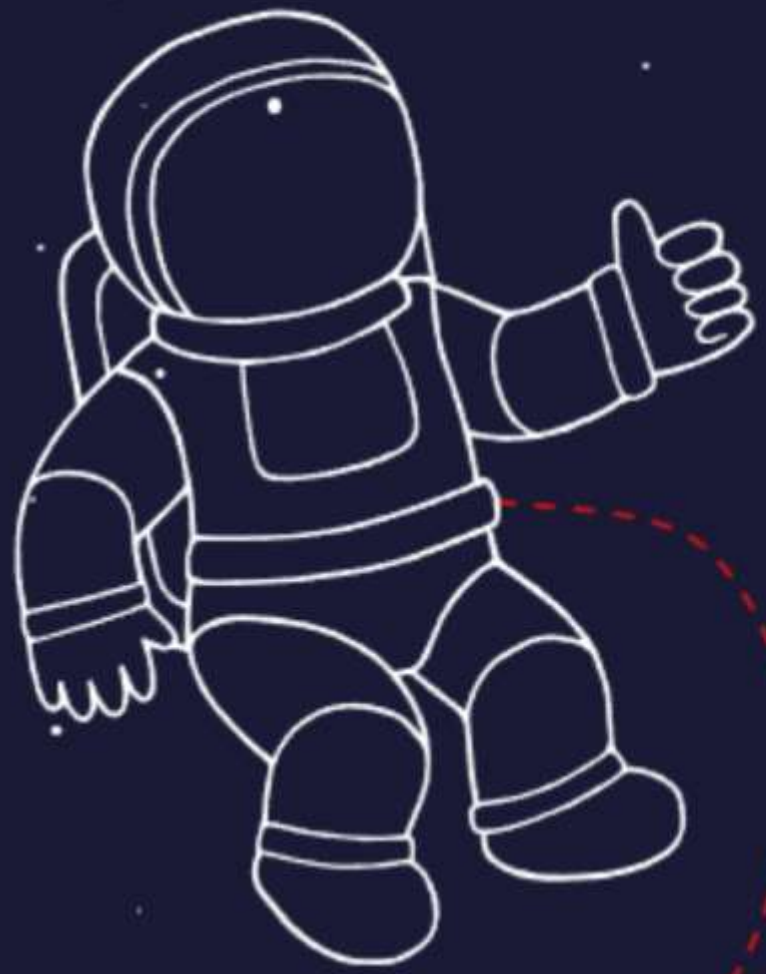
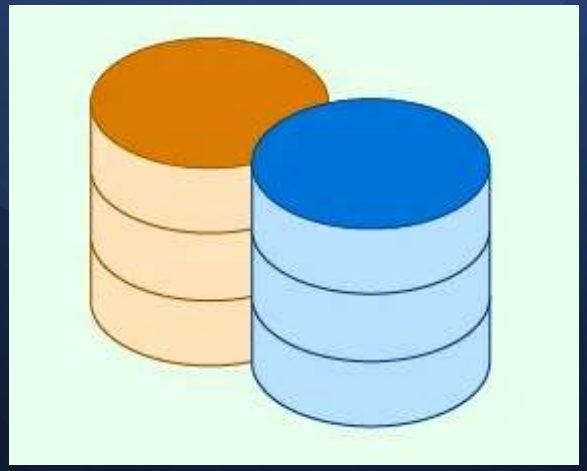


Programa académico CAMPUS

Trainer
Ing. Carlos H. Rueda C.

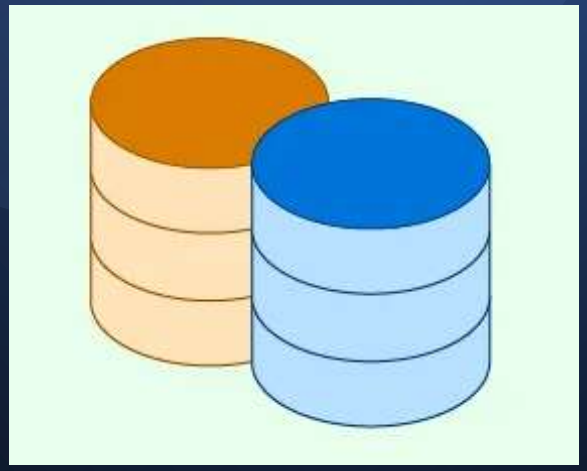
JAVASCRIPT LOCALSTORAGE

JAVASCRIPT LOCALSTORAGE



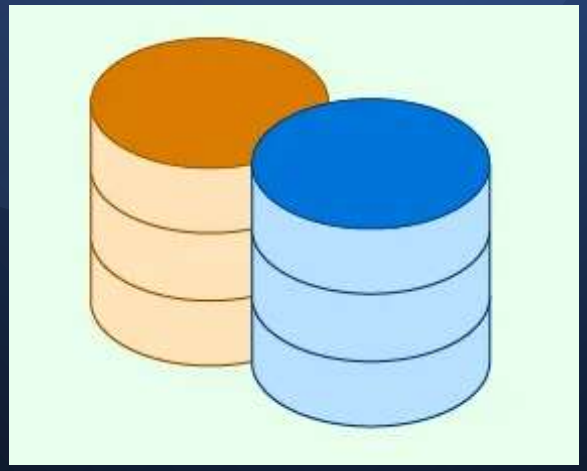
El objeto **Storage** (API de almacenamiento web) nos permite almacenar datos de manera local en el navegador y sin necesidad de realizar alguna conexión a una base de datos.

JAVASCRIPT LOCALSTORAGE



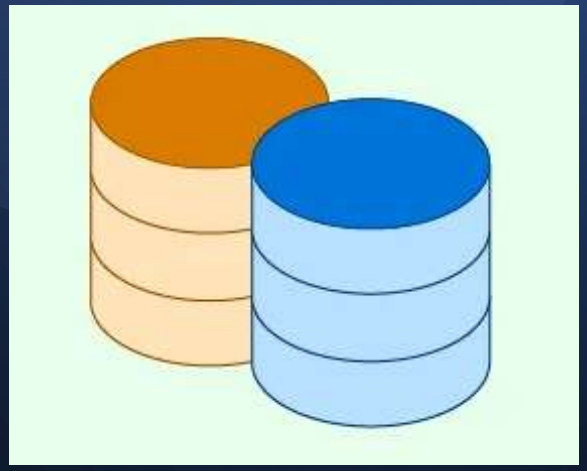
localStorage es una propiedad que accede al objeto **Storage** y tienen la función de **almacenar datos de manera local** hasta que se decida limpiar los datos del navegador.

JAVASCRIPT LOCALSTORAGE



- ✓Hasta **5 MB** por dominio en la mayoría de los navegadores.
- ✓Los **datos permanecen** incluso si el *navegador se cierra y se vuelve a abrir*.
- ✓Ideal para guardar configuraciones, **preferencias del usuario o pequeños datos** que no cambian mucho con el tiempo.
- ✓Es **sincrónico**, por lo que puede bloquear la interfaz si se usa con grandes cantidades de datos.

JAVASCRIPT LOCALSTORAGE

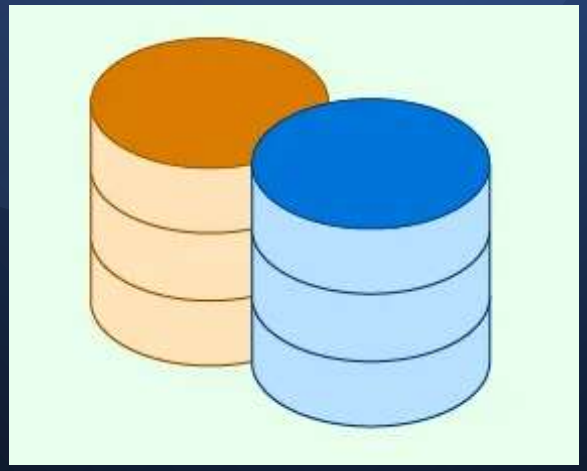


LocalStorage provee los siguientes métodos y propiedades:

setItem(clave, valor) – almacenar clave/valor.
getItem(clave) – obtener el valor
removeItem(clave) – eliminar la clave y valor.
clear() – borrar todo.
key(índice) – obtener la clave de una posición dada.
length – el número de ítems almacenados.

JAVASCRIPT LOCALSTORAGE

Validar objeto Storage en el navegador

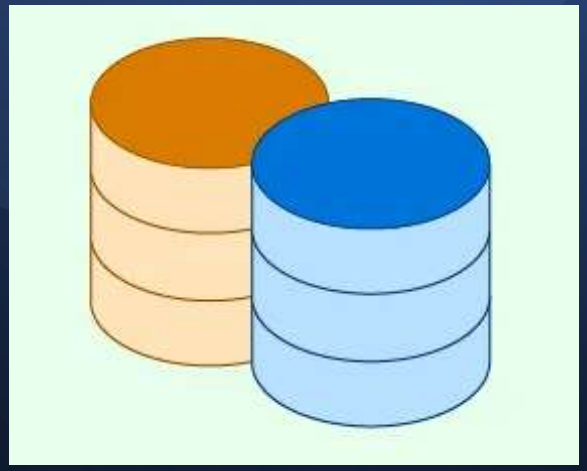


Los navegadores actuales soportan el objeto Storage, pero no está de más validar que realmente se pueda usar.

```
if (typeof(Storage) !==
'undefined') {
    // Código cuando Storage es
compatible
} else {
   // Código cuando Storage NO es
compatible
}
```

JAVASCRIPT LOCALSTORAGE

Guardar datos



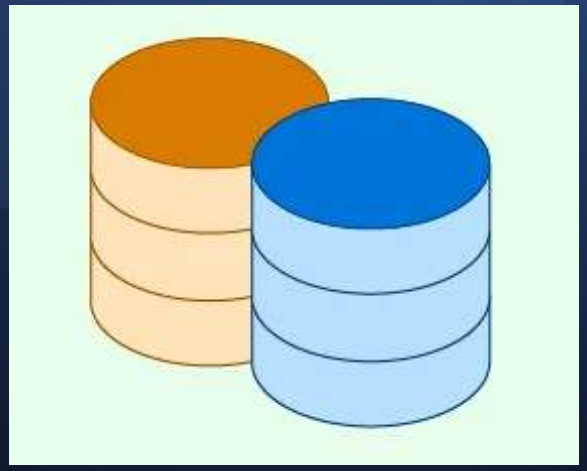
Hay dos formas de guardar datos:

- Acceso tipo Mapa
- Acceso tipo objeto

```
//acceso tipo mapa
localStorage.setItem('Nombre',
'Miguel Antonio')

//acceso tipo objeto
localStorage.Apellido = 'Márquez
Montoya'
```

JAVASCRIPT LOCALSTORAGE

Recuperar datos



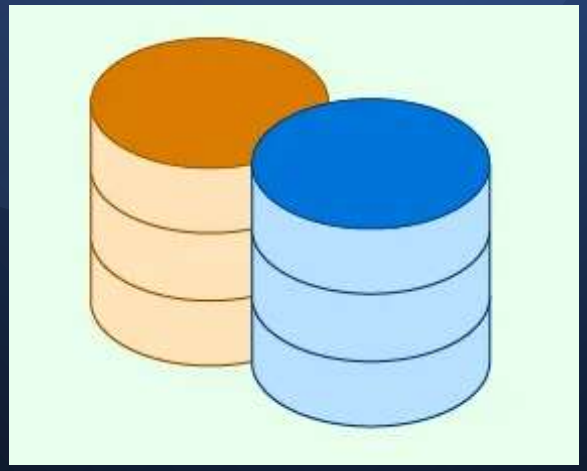
Hay dos formas de recuperar datos:

- Acceso tipo Mapa
- Acceso tipo objeto

```
// Op 1 -> localStorage.getItem(name, content)
// Op 2 -> localStorage.name


// Obtenemos los datos almacenados
let firstName = localStorage.getItem("Nombre"),
Let lastName = localStorage.Apellido;


console.log(`Hola, mi nombre es ${firstName}
${lastName}`);
// Imprime: Hola, mi nombre es Miguel Antonio
Márquez Montoya
```

JAVASCRIPT LOCALSTORAGE

Eliminar datos

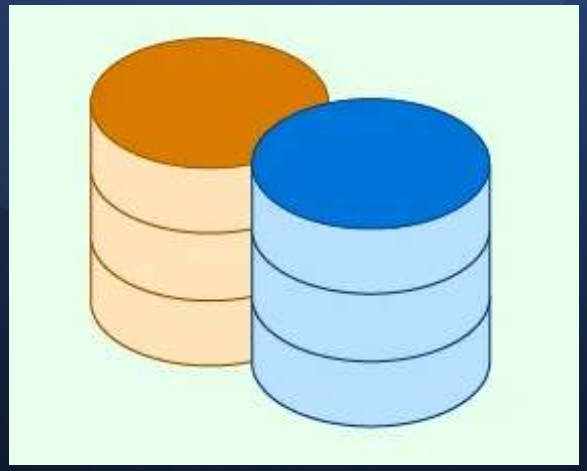


Para eliminar hacemos:

```
// localStorage.removeItem(name)
localStorage.removeItem(Apellido)
```

JAVASCRIPT LOCALSTORAGE

Limpiar todos los datos

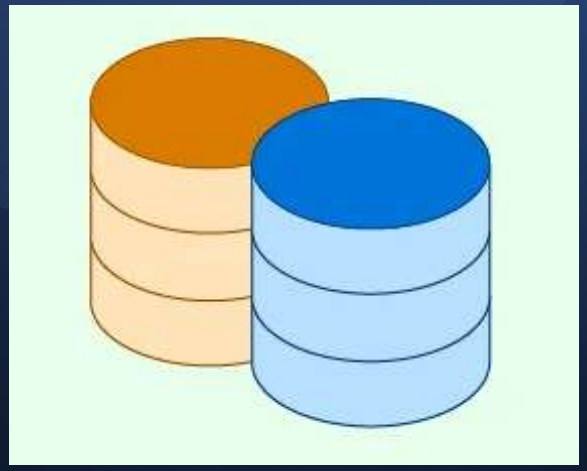


Dejar todo el objeto limpio

```
// localStorage.clear()
localStorage.clear()
```

JAVASCRIPT LOCALSTORAGE

Iterando el LocalStorage

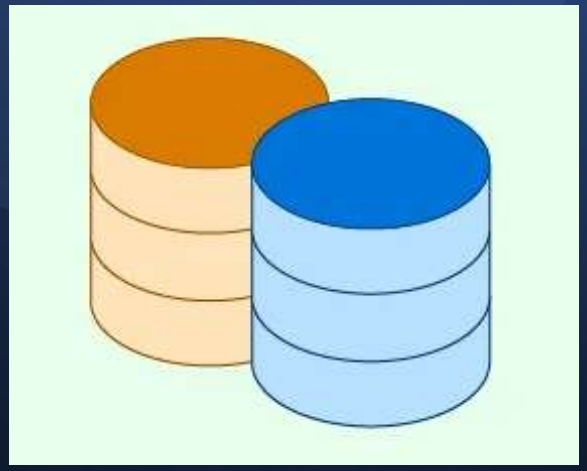


Una opción es utilizar iteración sobre un array:

```
/*Obtener datos almacenados*/
for(let i=0; i<localStorage.length; i++) {
    let key = localStorage.key(i);
    alert(`${key};
    ${localStorage.getItem(key)}`);
}
```

JAVASCRIPT LOCALSTORAGE

Iterando el LocalStorage

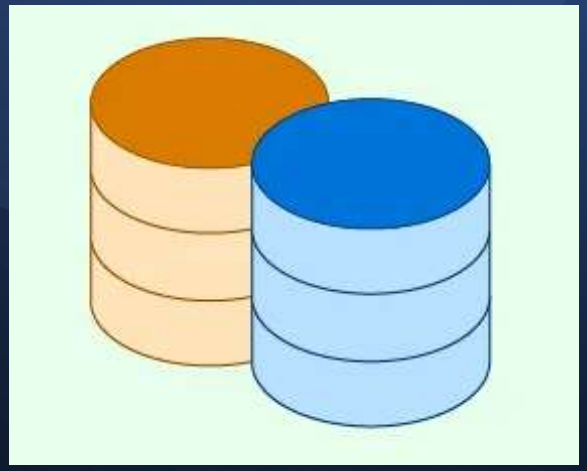


La forma anterior devuelve campos propios del localStorage que no necesitamos. Para solucionar esto podríamos filtrar los campos con hasOwnProperty

```
for(let key in localStorage) {
  if (!localStorage.hasOwnProperty(key)) {
  // se salta claves como "setItem", "getItem"
      continue;
   }
   alert(`${key}: ${localStorage.getItem(key)}`);
}
```

JAVASCRIPT LOCALSTORAGE

Iterando el LocalStorage

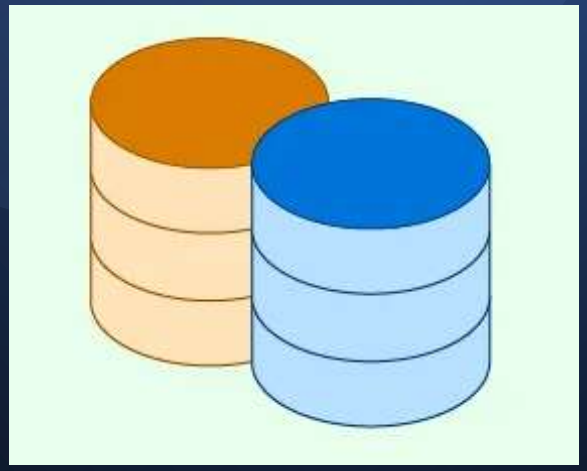


O podemos acceder a las "claves" propias con **Object.Key.**

```
let keys = Object.keys(localStorage);
for(let key of keys) {
   alert(`${key}: ${localStorage.getItem(key)}`);
}
```

JAVASCRIPT LOCALSTORAGE

Ejercicio



Ejemplo - localStorage

Nombre

Apellido

Guardar

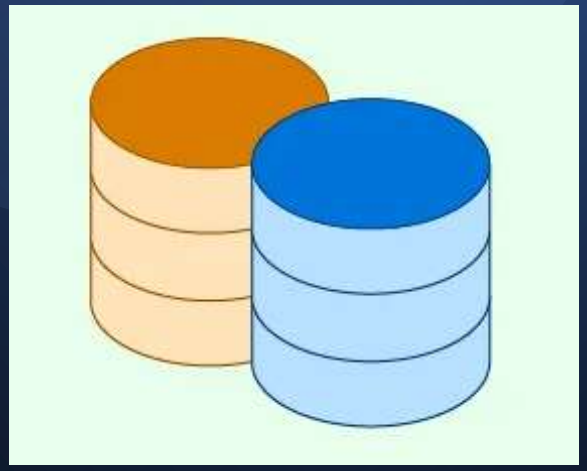
Nombre almacenado: Daniel

Apellido almacenado: Santos

Cargar elementos

JAVASCRIPT LOCALSTORAGE

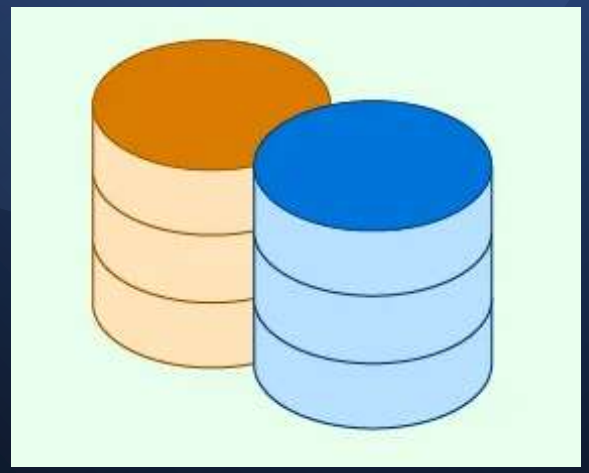
Ejercicio



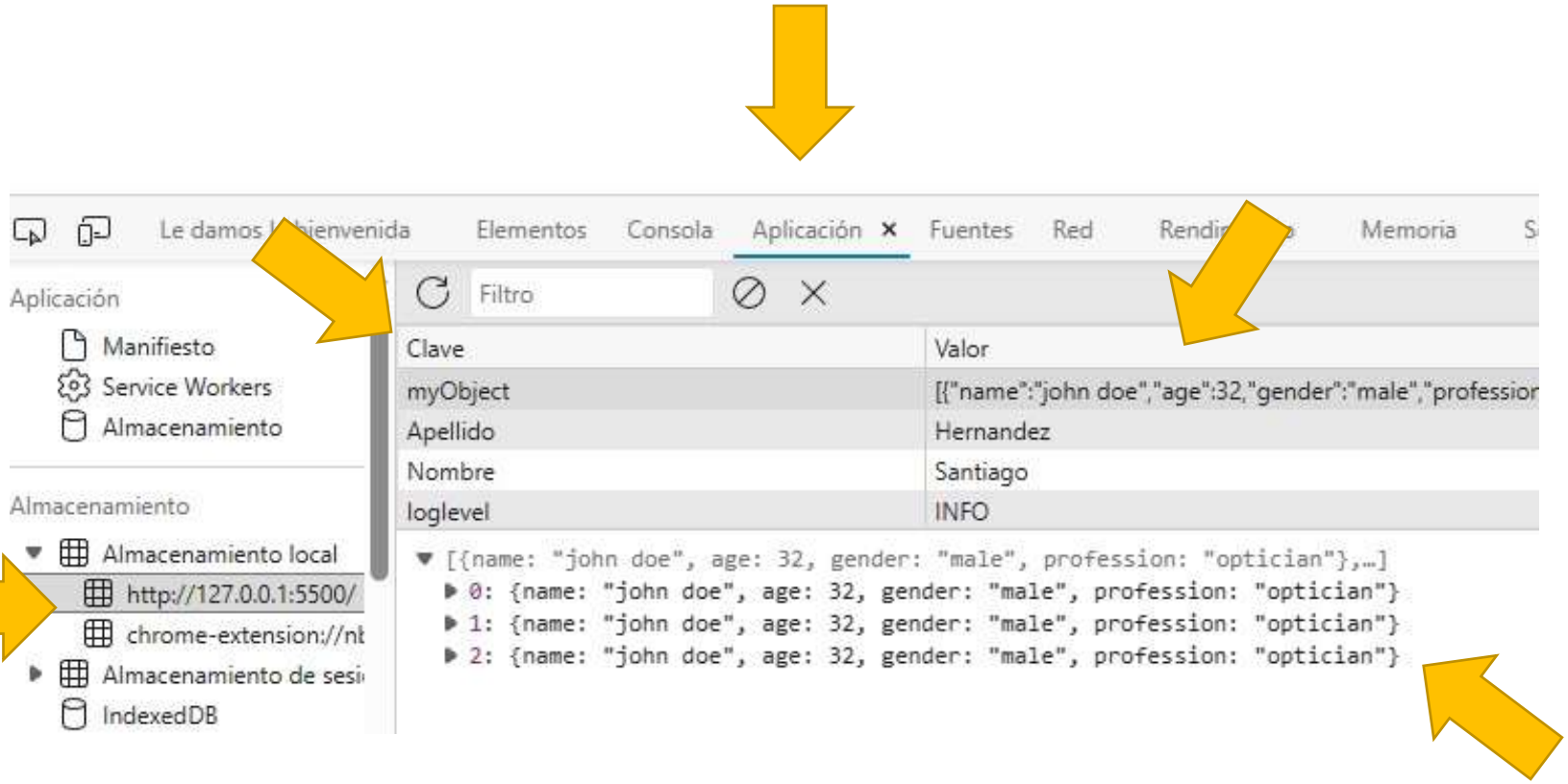
```
const myObject = {
          name : "john doe",
          age : 32,
          gender : "male",
          profession : "optician"
}

localStorage.setItem("myObject",
JSON.stringify(myObject));


let newObject =
window.localStorage.getItem("myObject");
console.log(JSON.parse(newObject));
```

JAVASCRIPT SESSIONSTORAGE

Pestaña Aplicación



Ejemplo localStorage



JAVASCRIPT SESSIONSTORAGE

SessionStorage almacena información mientras la pestaña donde se esté utilizando siga abierta, una vez cerrada, la información se elimina.

JAVASCRIPT SESSIONSTORAGE

Las propiedades y métodos son los mismos, pero es mucho más limitado:

- ✓ **sessionStorage** solo existe dentro de la pestaña actual del navegador.
  - Otra pestaña con la misma página tendrá un almacenaje distinto.
- ✓ Los datos sobreviven un refresco de página, pero no cerrar/abrir la pestaña.

JAVASCRIPT SESSIONSTORAGE

- ✓ Almacena datos en **formato clave-valor**
- ✓ Ofrece un límite de entre **5 MB** por dominio según el navegador
- ✓ Solo permite el acceso a scripts del mismo origen
- ✓ Los datos se eliminan al cerrar la pestaña o ventana del navegador
- ✓ **Sincrónico** como `localStorage`
- ✓ Ideal para almacenar **datos temporales,** como formularios parcialmente completados o **información** de navegación **entre páginas en una sesión**.

JAVASCRIPT SESSIONSTORAGE

Métodos

Método	Descripción
`setItem(key, value)`	Guarda un par clave-valor en el almacenamiento.
`getItem(key)`	Recupera el valor asociado a una clave específica.
`removeItem(key)`	Elimina un par clave-valor según la clave dada.
`clear()`	Borra todos los datos almacenados en `sessionStorage`.
`length`	Devuelve la cantidad de pares clave-valor almacenados.
`key(index)`	Recupera la clave en una posición específica dentro del almacenamiento.

JAVASCRIPT SESSIONSTORAGE

Ejemplos guardar datos

```
// Guardar datos del formulario en sessionStorage
document.getElementById("saveForm").addEventListener("click", () => {
    const formData = {
        name: document.getElementById("name").value,
        email: document.getElementById("email").value,
    };
    sessionStorage.setItem("formData", JSON.stringify(formData));
    alert("Datos guardados temporalmente en sessionStorage.");
});
```

JAVASCRIPT SESSIONSTORAGE

Ejemplos recuperar datos

```
// Recuperar datos al recargar la página
window.addEventListener("load", () => {
    const savedData = sessionStorage.getItem("formData");
    if (savedData) {
        const { name, email } = JSON.parse(savedData);
        document.getElementById("name").value = name;
        document.getElementById("email").value = email;
        console.log("Datos recuperados de sessionStorage:", { name, email });
    }
});
```

JAVASCRIPT SESSIONSTORAGE

Ejemplos Limpiar datos

```
// Recuperar datos al recargar la página
window.addEventListener("load", () => {
    const savedData = sessionStorage.getItem("formData");
    if (savedData) {
        const { name, email } = JSON.parse(savedData);
        document.getElementById("name").value = name;
        document.getElementById("email").value = email;
        console.log("Datos recuperados de sessionStorage:", { name, email });
    }
});
```