

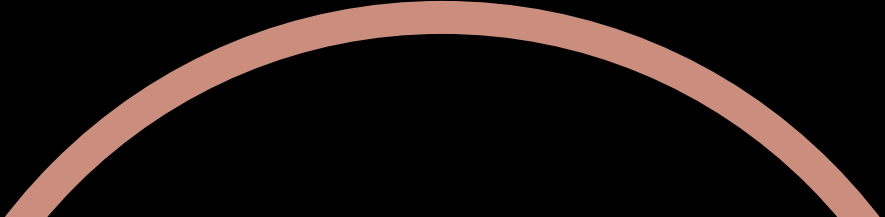


Tarea 2.

ALMACENAMIENTO JS

LIZZY PAMELA M.

IS-410



Introducción

¿ALMACENAMIENTO DE LADO DEL CLIENTE?

El almacenamiento de lado del cliente funciona con principios similares, pero tiene diferentes usos. Consiste en una API de JavaScript que te permiten almacenar datos en el cliente (es decir, en la máquina del usuario) y luego recuperarlos cuando sea necesario.

A menudo, el almacenamiento de lado del cliente y de lado del servidor se utilizan juntos. Por ejemplo, puedes descargar un lote de archivos de música (quizás utilizados por un juego web o una aplicación de reproducción de música), almacenarlos dentro de una base de datos de lado del cliente y reproducirlos según sea necesario. El usuario solo tendría que descargar los archivos de música una vez; en las visitas posteriores, se recuperarían de la base de datos.

Formas típicas de almacenar datos con JavaScript



Static storage



Cookies



sessionStorage



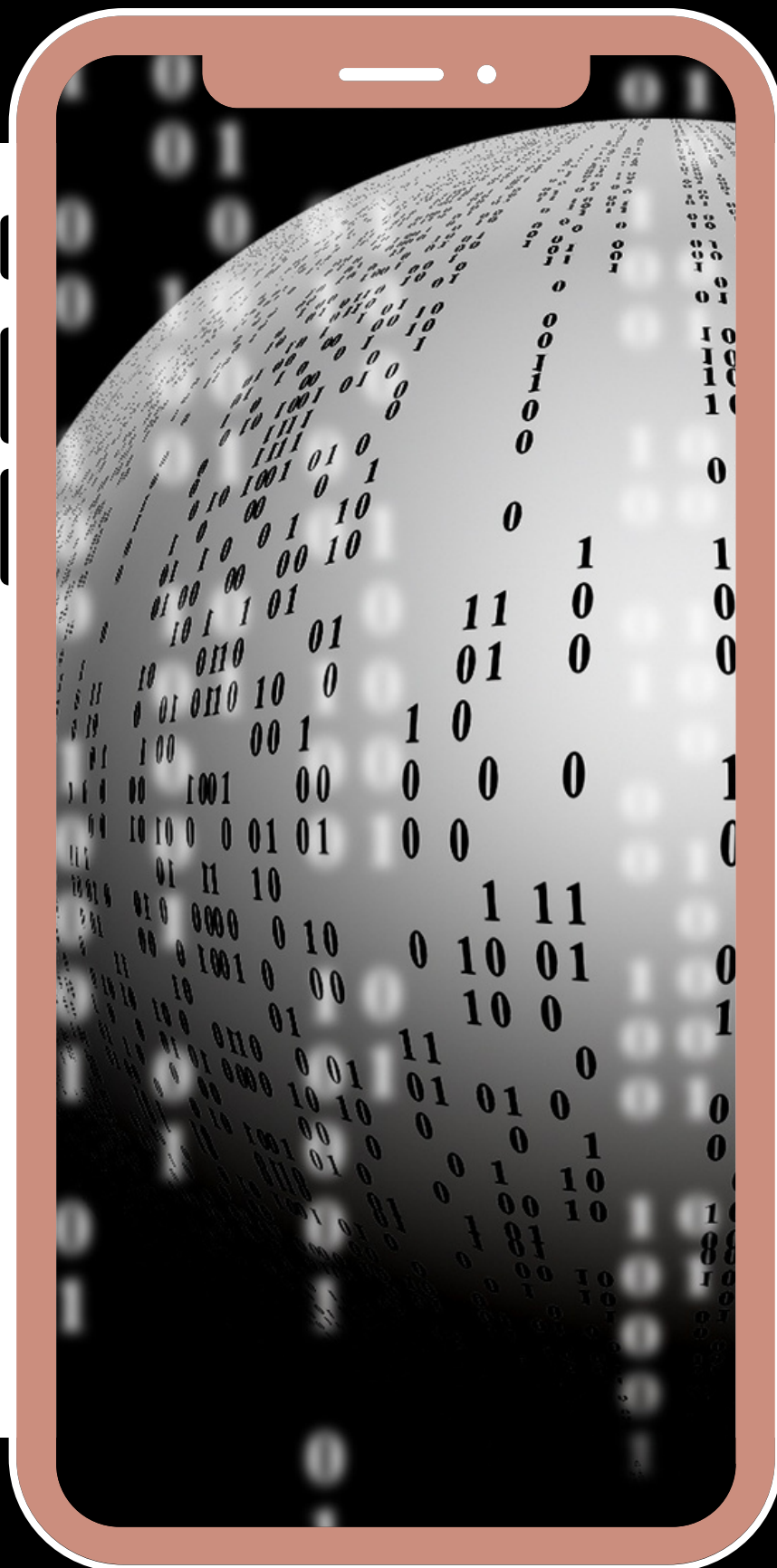
localStorage



Usando la base de datos del navegador (IndexedDB)



WebSQL



Como utilizar cookies en JavaScript

Una cookie es un string (cadena de texto) que contiene parejas parametro=valor separadas por ; de la siguiente forma:

```
<NOMBRE>=<VALOR>; EXPIRES=<FECHA>; MAX-AGE=<SEGUNDOS>; PATH=<RUTA>; DOMAIN=<DOMINIO>;  
SECURE; HTTPONLY;
```

HttpOnly: Opcional. Parámetro no disponible en JavaScript ya que crea cookies válidas sólo para protocolo HTTP/HTTPS y no para otras APIs, incluyendo JavaScript.

La propiedad *document.cookie* es todo lo que se necesita para trabajar con cookies client-side desde JavaScript. A través de ella podemos crear, leer, modificar y eliminar cookies.

LocalStorage, sessionStorage

Los objetos de almacenaje web localStorage y sessionStorage permiten guardar pares de clave/valor en el navegador

Lo que es interesante sobre ellos es que los datos sobreviven a una recarga de página (en el caso de sessionStorage) y hasta un reinicio completo de navegador (en el caso de localStorage). Lo veremos en breve.

Ambos objetos de almacenaje proveen los mismos métodos y propiedades:

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

Puntos clave:

- Tanto la clave como el valor deben ser strings.
- El límite es de más de 5mb+, dependiendo del navegador.
- No expiran.
- Los datos están vinculados al origen (dominio/puerto/protocolo).

IndexedDB



La API IndexedDB (a veces abreviada IDB) es un sistema de base de datos completo disponible en el navegador en el que puedes almacenar datos complejos relacionados, tipos de los cuales no se limitan a valores simples como cadenas o números. Puedes almacenar videos, imágenes y casi cualquier otra cosa en una instancia de IndexedDB.

Sin embargo, esto tiene un costo: IndexedDB es mucho más complejo de usar que la API de almacenamiento web

Aquí tienes los pasos básicos para declarar una base de datos IndexedDB en JavaScript:

1. Abrir o crear una base de datos: Utiliza el método **indexedDB.open()** para abrir una base de datos existente o crear una nueva si no existe.
2. **Manejar eventos de éxito y error:** Escucha los eventos **onsuccess** y **onerror** para manejar el éxito y el fallo de la operación de apertura de la base de datos.
3. **Definir el esquema de la base de datos:** En el evento **onupgradeneeded**, define la estructura de tu base de datos creando o modificando objetos de almacenamiento (object stores).
4. **Agregar y manejar objetos de almacenamiento:** Una vez que la base de datos esté abierta, puedes agregar, recuperar, actualizar y eliminar objetos en los objetos de almacenamiento.

JSON.stringify()

```
// Definimos un objeto JavaScript
var persona = {
  nombre: "Juan",
  edad: 30,
  ocupacion: "Ingeniero"
};

// Convertimos el objeto a una cadena JSON
var cadenaJSON = JSON.stringify(persona);

// Mostramos la cadena JSON resultante
console.log(cadenaJSON);
```

json

```
{"nombre":"Juan","edad":30,"ocupacion":"Ingeniero"}
```

Es un método en JavaScript que convierte un objeto JavaScript en una cadena de texto JSON. Este método es comúnmente utilizado cuando necesitas enviar datos estructurados a través de una red o almacenarlos en un archivo.

Esta cadena JSON puede ser transmitida a través de una red, almacenada en un archivo, o usada de otras maneras según tus necesidades. **JSON.stringify()** también puede ser útil para depurar y lograr información sobre el contenido de un objeto JavaScript.

WebSQL



WebSQL fue una especificación de API para bases de datos SQL basada en SQLite, que permitía a las aplicaciones web almacenar datos en una base de datos local en el navegador. Sin embargo, WebSQL ha sido desaprobado por la mayoría de los navegadores y ya no se recomienda su uso.

La especificación de la base de datos Web SQL está obsoleta desde noviembre de 2010. No se anima a los proveedores de navegadores a soportar esta tecnología y es importante que cualquiera que lea esto lo entienda.

En resumen, debido al abandono, la falta de soporte, las limitaciones de implementación y las alternativas más modernas disponibles, se recomienda evitar el uso de WebSQL y en su lugar utilizar IndexedDB para el almacenamiento de datos en el navegador.




```

state={
  products: storeProducts
}
render() {
  return (
    <React.Fragment>
      <div className="py-5">
        <div className="containe
          <Title name="our" ti
          <div className="row"
            <ProductConsumer
              {(value) => .
                console
            }}
          </ProductConsumer
        </div>
      </div>
    </React.Fragment>
  )
}

```

Gracias

"El aprendizaje es un tesoro que sigue su dueño en todas partes." - Proverbio chino