APUNTES DEL CURSO DE JavaScript

Almacenar textos con Local Storage

Derimán Tejera Fumero

20/08/2024

Código directo

```
//? ============================ Variables ============================
const listaTweets = document.querySelector("#lista-tweets");
const formulario = document.querySelector("#formulario");
let tweets = [];

//? ========================= Event Listeners =========================
eventListeners();

function eventListeners() {
  // Cuando se envia el formulario
  formulario.addEventListener("submit", agregarTweet);

  // Borrar Tweets
  listaTweets.addEventListener("click", borrarTweet);

  // Contenido cargado
  document.addEventListener("DOMContentLoaded", () => {
    // Si el resultado de JSON.parse( ... ) es null o undefined (lo que ocurre si no hay nada almacenado bajo la clave "tweets"),
    // se asigna un array vacío ([]) a la variable tweets.
    // || [], se asegura que la variable tweets siempre tenga un valor útil (un array), incluso si no hay nada en localStorage.
    // Esto permite que el código que sigue pueda funcionar de manera consistente sin tener que comprobar si tweets es null o undefined.
    tweets = JSON.parse(localStorage.getItem("tweets")) || [];
    console.log(tweets);
    crearHTML();
  });
}

// Añadir tweet del formulario
function agregarTweet(e) {
  e.preventDefault();
  // Lee el valor del textarea. En el caso de que sea un espacio vacío, no lo añadirá a la lista.
  const tweet = document.querySelector("#tweet").value.trim();

  // Validación
  if (tweet === "") {
    mostrarError("Un mensaje no puede ir vacio");
    return;
  }

  // Crear un objeto Tweet
  const tweetObj = {
    id: Date.now(),
    texto: tweet,
  };

  // Añadirlo a mis tweets
  tweets = [...tweets, tweetObj];

  // Una vez agregado, mandamos renderizar nuestro HTML
  crearHTML();

  // Reiniciar el formulario
  formulario.reset();
}

function mostrarError(error) {
  const mensajeEerror = document.createElement("p");
  mensajeEerror.textContent = error;
  mensajeEerror.classList.add("error");

  const contenido = document.querySelector("#contenido");
  contenido.appendChild(mensajeEerror);

  setTimeout(() => {
    mensajeEerror.remove();
  }, 3000);
}

function crearHTML() {
  limpiarHTML();

  if (tweets.length > 0) {
    tweets.forEach((tweet) => {
      // crear boton de eliminar
      const botonBorrar = document.createElement("a");
      botonBorrar.classList = "borrar-tweet";
      botonBorrar.innerText = "X";

      // Crear elemento y añadirle el contenido a la lista
      const li = document.createElement("li");

      // Añade el texto
      li.innerText = tweet.texto;

      // añade el botón de borrar al tweet
      li.appendChild(botonBorrar);

      // añade un atributo único...
      li.dataset.tweetId = tweet.id;

      // añade el tweet a la lista
      listaTweets.appendChild(li);
    });
  }

  sincronizarStorage();
}

// Elimina el Tweet del DOM
function borrarTweet(e) {
  e.preventDefault();

  // console.log(e.target.parentElement.dataset.tweetId);
  const id = e.target.parentElement.dataset.tweetId;
  tweets = tweets.filter((tweet) => tweet.id !== id);
  crearHTML();
}

// Agrega tweet a local storage
function sincronizarStorage() {
  localStorage.setItem("tweets", JSON.stringify(tweets));
}

// Elimina los cursos del carrito en el DOM
function limpiarHTML() {
  while (listaTweets.firstChild) {
    listaTweets.removeChild(listaTweets.firstChild);
  }
}
```