**📑 Apuntes de contenidos por video**

**1. Intro. cómo ejecutar código**

Aprendemos cómo ejecutar código JavaScript en distintos entornos: desde la consola del navegador hasta el uso de editores como VSCode con Live Server.

**2. Variables JavaScript**

Exploramos cómo declarar variables con `var`, `let` y `const`, sus diferencias y en qué casos es recomendable usar cada una.

**3. Intro HTML y CSS**

Repaso básico de HTML para estructurar una página web, y CSS para darle estilo visual, colores, fuentes y organización.

**4. Funciones**

Se enseña a crear funciones para encapsular bloques de código reutilizables, incluyendo funciones tradicionales y funciones flecha.

**5. Unir HTML y JavaScript**

Vinculamos nuestros archivos HTML con JavaScript para permitir la interacción dinámica entre el usuario y la interfaz web.

**6. Botones**

Manejamos la interacción con botones: cómo detectar clics y asociarles funciones que se ejecuten al activarlos.

**7. If, objetos y fechas**

Utilizamos condicionales (`if/else`) para tomar decisiones, aprendemos a crear objetos con propiedades, y usamos el objeto `Date`.

**8. addEventListener vs onclick**

Comparación entre dos formas de manejar eventos. `addEventListener` permite una mejor separación del código y múltiples escuchas por evento.

**9. Inicio del proyecto de gestión económica**

Comenzamos a crear una aplicación para gestionar ingresos y egresos, aplicando lo aprendido hasta ahora en un proyecto real.

**10. Creando tabla en HTML**

Estructuramos una tabla en HTML para visualizar los datos registrados de manera ordenada y clara en filas y columnas.

**11. Name vs ID**

Diferencias clave: `id` sirve para identificar elementos de forma única, y `name` se utiliza especialmente en formularios para agrupar y enviar datos.

**12. preventDefault**

Método que se usa para evitar comportamientos predeterminados de elementos HTML, como recargar la página al enviar un formulario.

**13. FormData**

Utilizamos `FormData` para capturar y manipular fácilmente los datos enviados por un formulario en el navegador.

**14. Agregar datos a tabla desde un formulario**

Conectamos el formulario a la tabla: capturamos datos ingresados por el usuario y los mostramos dinámicamente como nuevas filas.

**15. Local Storage (base de datos del navegador)**

Introducción al `localStorage`, una forma de guardar datos en el navegador de manera persistente, incluso si el usuario cierra la página.

**16. Local Storage: almacenar un objeto en JSON**

Aprendemos a guardar objetos en `localStorage` convirtiéndolos a formato JSON con `JSON.stringify()`, y a recuperarlos como objetos válidos con `JSON.parse()`.

**17. Guardando datos de nuestra app en Local Storage**

Integramos el almacenamiento local con nuestra app para guardar ingresos y egresos. Así, los datos persisten aunque se cierre o recargue la página.

**18. Guardando el historial de transacciones en Local Storage (array de objetos)**

Guardamos un array de objetos representando cada transacción. Esto permite mantener un historial estructurado y fácilmente accesible desde el `localStorage`.

**19. Renderizar nuestra tabla desde el Local Storage al iniciar la app**

Automatizamos la carga de datos al iniciar la aplicación: la tabla se rellena dinámicamente con las transacciones guardadas previamente en `localStorage`.

**20. Limpiar un formulario con JavaScript - cómo resetear formulario**

Usamos el método `.reset()` para limpiar todos los campos de un formulario después de enviarlo, mejorando la experiencia del usuario y evitando duplicados.

**21. Los data atrributes en HTML y JS - guardar ifno en HTML**

Aprendimos a usar los data-\* attributes para almacenar información personalizada en los elementos HTML. Esto permite vincular datos específicos directamente en la interfaz, accesibles fácilmente desde JavaScript mediante element.dataset.

23. querySelector y los data attributes, combinación implacable

Combinamos querySelector con los data-attributes para seleccionar elementos de forma precisa y dinámica. Esta técnica facilita acceder a información embebida en HTML y manipular el DOM de manera eficiente e intuitiva.