Actividad 1: Botones con diferentes comportamientos

Descripción del problema:

El objetivo de esta actividad es crear una página web con botones que realicen acciones específicas al interactuar con ellos. Se requiere:

- Un botón llamado "Objeto mágico" que desaparece al hacer clic.
- Un botón denominado "Pulsador" que muestra un mensaje al pulsarlo.
- Dos botones deslizantes: uno que despliega/colapsa una lista de películas y otro que hace lo mismo con una lista de series.

Proceso de desarrollo:

- Creación de HTML: Se utilizan elementos <button> y listas desplegables <ul> para los submenús de películas y series.
- Estilo en CSS: Se aplican estilos básicos para que las listas estén ocultas por defecto.
- Implementación en JavaScript:
  - Se asignan eventos onclick a los botones para realizar las acciones deseadas.
  - Se usan condicionales en JavaScript para alternar la visibilidad de las listas y mostrar el mensaje en el botón "Pulsador".
  - Se añade console.log() para depuración y garantizar que la función se ejecute.

Funcionamiento:

- Botón "Objeto mágico": Al hacer clic, se ejecuta un evento onclick que cambia la propiedad display del botón a none, ocultándolo de la vista.
- Botón "Pulsador": Al pulsar, se muestra un alert() con el texto "Has pulsado sobre el botón".
- Botones de películas y series: Estos alternan la visibilidad de sus listas respectivas utilizando style.display para cambiar entre block y none.

Actividad 2: Tooltip dinámico

Descripción del problema:

El objetivo es implementar un sistema de tooltip que muestre el texto almacenado en un atributo data-tooltip al pasar el cursor sobre un elemento. Solo un tooltip debe aparecer a la vez.

Proceso de desarrollo:

- Estructura HTML: Se añaden elementos con el atributo data-tooltip que contiene el texto del tooltip.
- Estilo en CSS: Se define la clase .tooltip con propiedades de diseño como position, border, y box-shadow.

JavaScript:

- Se escucha el evento mouseover para detectar cuando el cursor pasa sobre un elemento con data-tooltip.
- Se crea dinámicamente un para el tooltip y se posiciona cerca del elemento que lo desencadenó.
- Se elimina el tooltip al salir del área con mouseout.

Funcionamiento:

- Al pasar el cursor sobre un elemento con data-tooltip, se crea y muestra un tooltip con el texto especificado.
- El tooltip sigue al cursor y desaparece al salir del área del elemento, asegurando que solo uno esté visible al mismo tiempo.

Actividad 3: Control deslizante

Descripción del problema:

El objetivo es crear un control deslizante que permita mover un pasador con el ratón y que este se detenga en los bordes del slider. El control debe funcionar incluso si el ratón se mueve rápidamente fuera de la barra.

Proceso de desarrollo:

Diseño en HTML y CSS:

- Se define una estructura HTML con un contenedor para el slider y un elemento para el pasador (.thumb).
- Se aplican estilos básicos al slider y al pasador.

JavaScript:

- Se utiliza el evento mousedown para detectar cuando el pasador es presionado.
- Se manejan los eventos mousemove y mouseup para controlar el arrastre y liberar el pasador.
- Se limita el movimiento del pasador al área del slider.

Funcionamiento:

- El pasador puede moverse horizontalmente al arrastrarlo con el ratón.
- El movimiento se detiene al llegar a los bordes del slider gracias a la verificación de límites en el código.
- Se asegura que el arrastre sea fluido y que el pasador se detenga al soltar el botón del ratón.

Actividad 4: Arrastre de elementos

Descripción del problema: Se requiere que los elementos con la clase draggable puedan ser arrastrados por la página, pero sin salir del área visible de la ventana. La página debe desplazarse automáticamente si el elemento alcanza el borde superior o inferior de la ventana.

Proceso de desarrollo:

HTML y CSS:

- Se definen elementos con la clase draggable.
- Se aplican estilos básicos para identificar visualmente los elementos arrastrables.

JavaScript:

- Se escucha el evento mousedown en todo el documento para detectar cuándo se inicia el arrastre.
- Se utilizan los eventos mousemove y mouseup para manejar el arrastre y detenerlo al soltar el ratón.
- Se implementan restricciones para que los elementos no se salgan de la ventana.

Funcionamiento:

- Al hacer clic y arrastrar un elemento con la clase draggable, se sigue el movimiento del ratón dentro de los límites de la ventana.
- La página se desplaza automáticamente si el elemento se mueve hacia el borde superior o inferior, permitiendo un arrastre continuo.
- El arrastre se detiene al soltar el botón del ratón (mouseup), y los elementos no se pueden mover fuera del área visible.