Desarrollo Web – Resumen de estudio

# Notas de JS (Java Script)

Programamos la lógica del juego Mokepon que básicamente es elegir el Mokepon con un botón de seleccionar, automáticamente se elige de manera aleatoria el mokepon enemigo.

Por otra parte, la sección de ataque elegimos con botones fuego, agua y tierra de nuevo el enemigo realiza ataques aleatorios con las mismas

Se restan las vidas tanto del jugador como del enemigo, el juego termina cuando uno de los dos ya no tenga vidas, en la pantalla muestra los ataques de ambos y mencionan si el jugador gano o perdió

Al llegar a la vida en 0 de uno de ellos muestra el mensaje de juego ganado o perdido

El botón de reinicio se muestra al finalizar el juego para volver a empezar

## Funciones:

1. Iniciar Juego
2. Crear Mensaje
3. Combate
4. Revisar vidas
5. Crear mensaje final
6. Enemigo atacando
7. Ataques, fuego tierra agua
8. Aleatorio
9. Seleccionar mascota del jugador
10. Seleccionar mascota del enemigo
11. Reiniciar juego

## Códigos aprendidos

* **Document.getElementById:** se selecciona la etiqueta con el “id” correspondiente para guardarlo en una variable (let) y de esa manera acceder a sus funciones
* **VariableEtiqueta.style.display = “none” o “block”:** oculta o muestra (en bloque) en este caso la sección (forma JS de hacerlo porque se puede hacer con CSS
* **VariableBotonAsignado.addEventListener(“acción”, función):** De esta forma se crea un evento de escucha, para el elemento indicado previamente en la variable guardada, que al ocurrir el evento una función se dispara
* **Element = document.createElement (“”):** previo a esto se tuvo que asignar en otra variable diferente a esta, la sección a ser editada, porque dicha función asigna un elemento dentro de las comillas y luego invocando una función de “variableElement” *innerHTML y luego se agrega con appenChild*
* **innerHTML:** se agrega contenido dentro del contenedor, sobre escribe lo existente dentro de DOM
* **appendChild:**  Agrega un nuevo nodo al final de la lista de un elemento hijo de un elemento padre especificado.
* **window.addEventListener("load", iniciar Juego):** EL script fue invocado dentro del head entonces esto agrega un evento de escucha que; una vez cargado todo el documento invoca la función iniciar juego que crea los otros escuchadores para empezar con la interacciones

# Notas de HTML

### Elementos aprendidos

* <script></script> = Para referenciar el script la lógica del JavaScript
* <section></section> = El elemento section es un elemento de seccionamiento genérico, diseñado para contener una parte de un documento temáticamente definida. Los contenidos de un elemento section son usualmente precedidos por un encabezado.
* <div> = Sirve para crear secciones o agrupar contenidos.
* <input type="radio" name="mascota" id="hipodoge"> = se usa para crear controles interactivos para formularios basados en la web con el fin de recibir datos del usuario
  + **Name:** Los botones radio que tengan el mismo valor para su atributo name están dentro del mismo "grupo de botones radio". Solo un botón radio dentro de un grupo puede ser seleccionado a la vez.
  + **Radio:** es el atributo del input puede contener mas tipos de inputs
* <label> = representa una etiqueta para un elemento en una interfaz de usuario
  + For: El ID del elemento de formulario etiquetable *(id=hipodoge)* relacionado en el mismo documento que el elemento label. El primer elemento en el documento con tal ID que coincida con el dispuesto en el atributo for será el control etiquetado para este elemento.
  + Parafraseando, el <input> y <label>, están relacionados por medio del ID del intput y el for del label, para asociar el texto con el radio ya que el radio es muy chiquito para acertar

# Notas de CSS

### Anatomía de CSS

* Tipo = El nombre de la etiqueta HTML tal cual (h1, p , section , body, etc)
* Clase = El nombre que le dimos a la clase en HTML simplemente le agregamos (.) punto al principio.
* ID = El nombre que le dimos al id en HTML simplemente le agregamos (#) hashtag al principio. Cada una se debe usar en casos específicos pero creo que eso lo veremos más adelante.

### Tipo de visualización (Display)

Block Inline

Texto, Correo electrónico

Descripción generada automáticamente con confianza mediaImagen que contiene texto, firmar, señal, tráfico

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente con confianza mediaInline – block

### CSS Flexbox

Una herramienta para poder alinear los elementos de manera mas ordenada

Se puede ordenar de forma vertical o de forma horizontal no ambos

Una captura de pantalla de un celular con letras

Descripción generada automáticamente con confianza media Una captura de pantalla de un celular con letras

Descripción generada automáticamente con confianza media

### Solo el hijo inmediato

Una cosa MUY importante que quiero señalar es que FlexBox solo envuelve a sus hijos inmediatos. El FlexBox no se envuelve más allá de una capa de profundidad. Sólo los hijos inmediatos. Entonces NO hay una relación de nietos o nietos. ¡Solo padre ↔️ Hijos inmediatos!

Por supuesto, puede establecer un Flexbox siempre que exista una relación padre-hijo. Entonces, un niño también puede ser el FlexBox para sus hijos. Pero será un FlexBox separado. Y no transfiere las propiedades flexibles de los abuelos.