

Contenido

[**I. INTRODUCCIÓN** 3](#_heading=h.gjdgxs)

[**II. DESARROLLO**](#_heading=h.30j0zll) 5

[**ESTRUCTURA DE LA APLICACIÓN**](#_heading=h.3znysh7)5

**HTML** 6

**CSS**8

**JAVASCRIPT**10

[**RESULTADO FINAL**](#_heading=h.1t3h5sf) 17

[**III. CONCLUSIONES** 1](#_heading=h.4d34og8)9

# INTRODUCCIÓN

Con la presente memoria de la actividad 1 asignatura Desarrollo de Aplicaciones y Servicios Web II lado del cliente (Front-end), En donde se aplicó el lenguaje de programación JavaScript y el lenguaje de marcado HTML y el lenguaje de estilos CSS, con la cual se pretende brindar al lector una explicación detallada de cada uno de los puntos analizados y desarrollados, de este modo se tenga claridad de los desarrollos realizados, imágenes de las diferentes vistas y posibilidades que la aplicación desarrollada ofrece.

A continuación, se hace un resumen de los requisitos solicitados en la actividad, para posteriormente adentrarnos en el desarrollo de la aplicación y la explicación de puntos clave para entender la misma.

De manera general, la presente aplicación tiene como objetivo la creación de un juego de cartas llamado solitario, el cual consiste en la colocación de 48 cartas de manera ordenada descendente desde la número 12 al número 1 en cuatro contenedores diferentes.



***Imagen 1.- Presentación juego solitario***

La baraja de cartas será una baraja VIU, que consiste en:

* 12 números 1,2,3,4,5,6,7,8,9,10,11 y 12.
* 4 palos: ”vius”, cuadrados, hexágonos, óvalos y círculos.



***Imagen 2.- Palos de la baraja***

Existirán 6 tapetes (superficies que albergarán mazos de cartas): El tapete inicial 1, el tapete de sobrantes 2, y cuatro tapetes receptores 3,4,5 y 6.

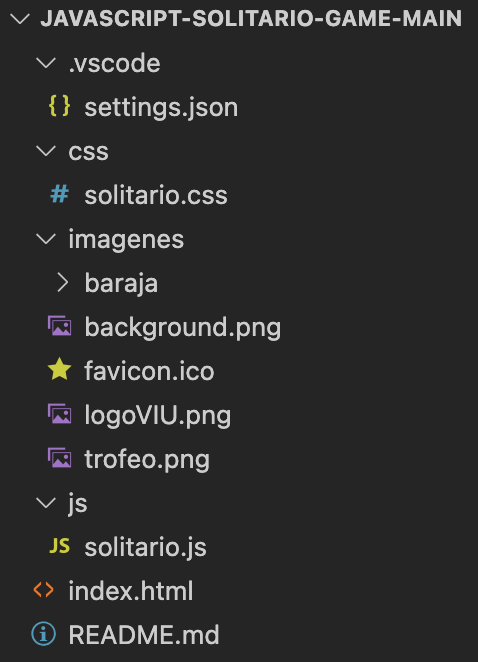


***Imagen 3.- Contenedores***

# DESARROLLO

## ESTRUCTURA DE LA APLICACIÓN

Se procedió al desarrollo de la aplicación, obteniendo como resultado la siguiente estructura de esta.



***Imagen 4.- Estructura de la aplicación***

A continuación, se explica el contenido de cada una de las carpetas y archivos encontrados al interior del archivo comprimido que acompaña la presente memoria.

* **Carpeta CSS**: Contiene el fichero solitario.css que define los estilos de este juego.
* **Carpeta js**: Contiene el fichero solitario.js que contiene toda la lógica del juego.
* **index.html**: Es el archivo con la estructura que tiene el proyecto.
* **README**: Es el archivo informativo que contiene los nombres de los integrantes, curso y las tecnologías usadas.
* **imagenes**: Esta carpeta contiene todas las imágenes utilizadas en el proyecto.

**HTML**

Las primeras líneas de código realizadas en el head se realizan definiendo las propiedades del título, descripción, tipo e imagen, además se implementaron algunas medidas de seguridad con la cabecera referrerpolicy=”no-referrer”.

Se agregó etiquetas para open graph creada por facebook que permite mostrar el contenido de la página como son el título, la descripción, la imagen de la página, etc.

og:title: se encarga de decirle al buscador de que se trata nuestra página.

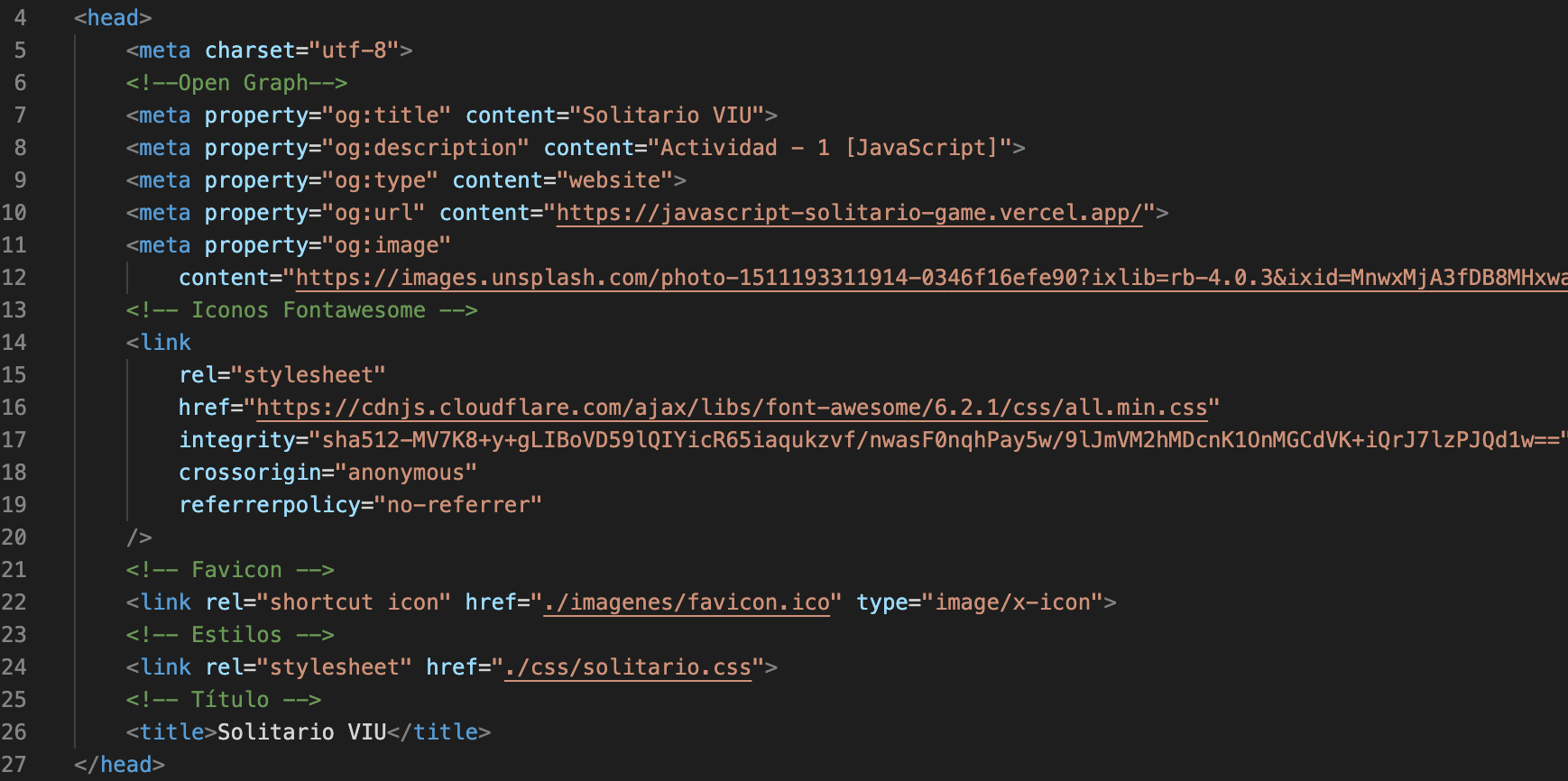
og:description: describe más detalladamente la información de la página.

og:url: muestra lo más limpiamente posible la URL única de cada página.

Se utilizó la librería de Fontawesome para insertar iconos que no permitieran mostrar un diseño mejorado.

Se agregó favicon para insertar la imagen del titulo que esta en la pestaña de nuestra página web.

Se incorpora el link de los estilos rel=”stylesheet”



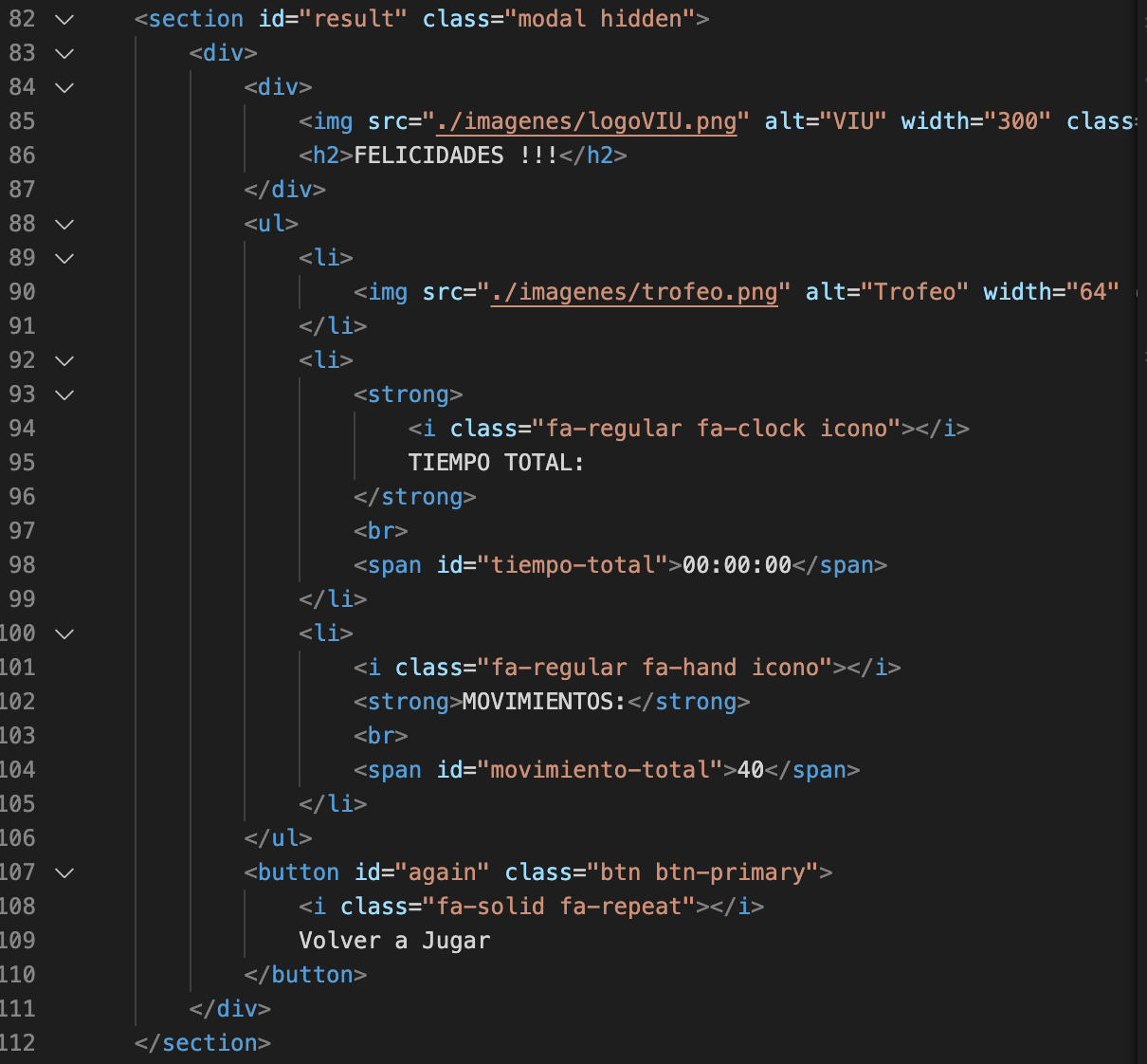
***Imagen 5.- Cabecera***

El body se divide en dos contenedores, el contenedor fade que es el que contiene todo el juego, en él se encuentra la baraja principal y las secundarias.



***Imagen 6.- Contenedor fade***

El contenedor modal hidden es el que contiene la ventana emergente en la cual se visualizará un mensaje de felicitación con los datos obtenidos en el transcurso del juego, como son: el tiempo total empleado y el número de movimientos realizados para conseguir finalizarlo, se agregan varios ID para capturarlo y poder trabajarlo utilizando JavaScript, además se agregan las clases para asignarle varios estilos.



***Imagen 7.- Contenedor modal hidden***

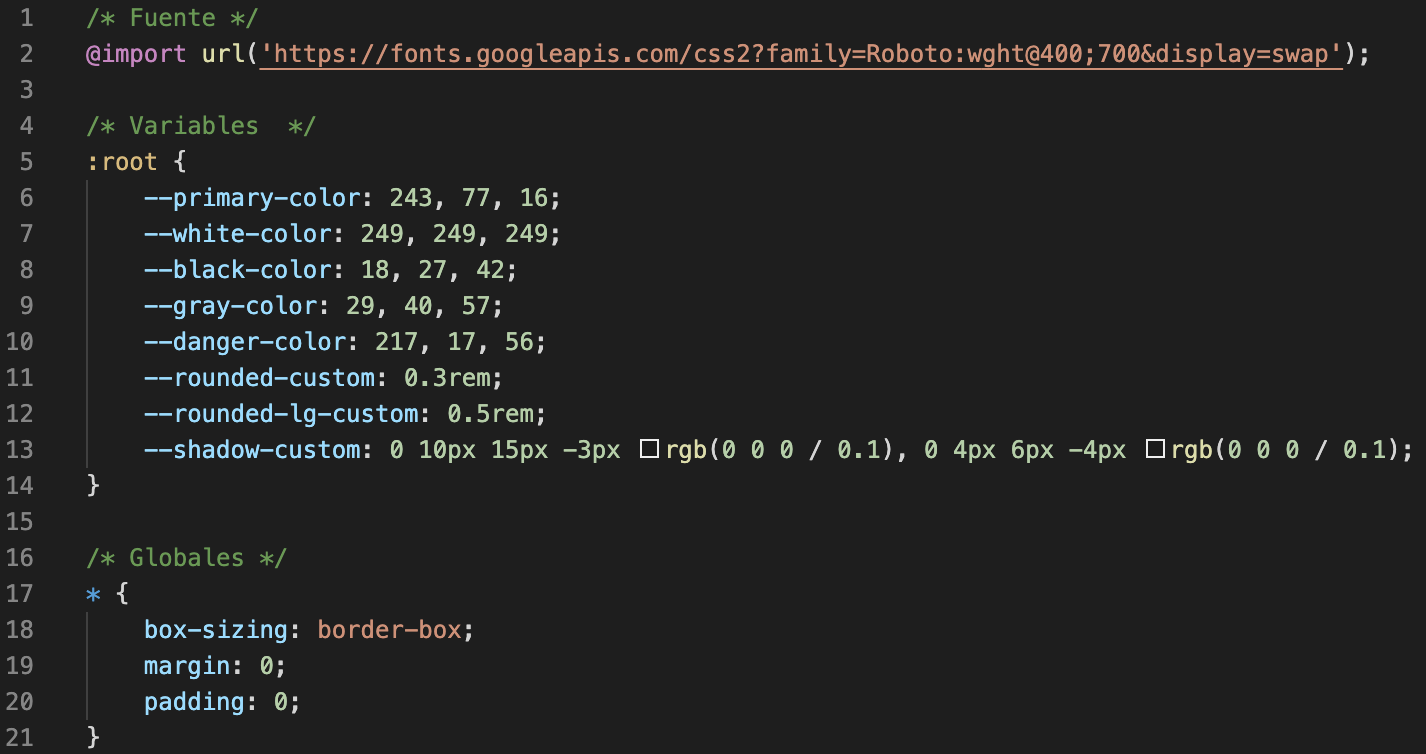
## 

## 

## CSS

Para comenzar se define la fuente utilizando el recurso de Google Fonts, recurso que permite acceder de forma gratuita a miles de fuentes tipográficas.

Se definen las variables, luego los estilos globales y colores del front para el juego del solitario en CSS.



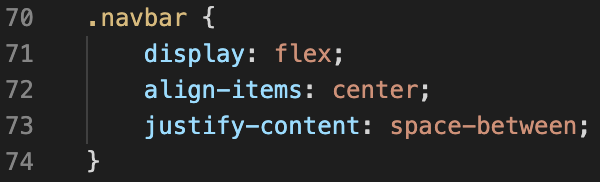
***Imagen 8.- Definición de variables***

Con la propiedad list-style: none estamos definiendo que no muestre ningún marcador en las listas de elementos, es decir que no muestre ningún tipo de viñeta, además se define la fuente que se importó previamente font-family: ‘Roboto’, sans-serif.



***Imagen 9.- Propiedad List-style***

Se definen estilos para la cabecera con la clase .navbar.



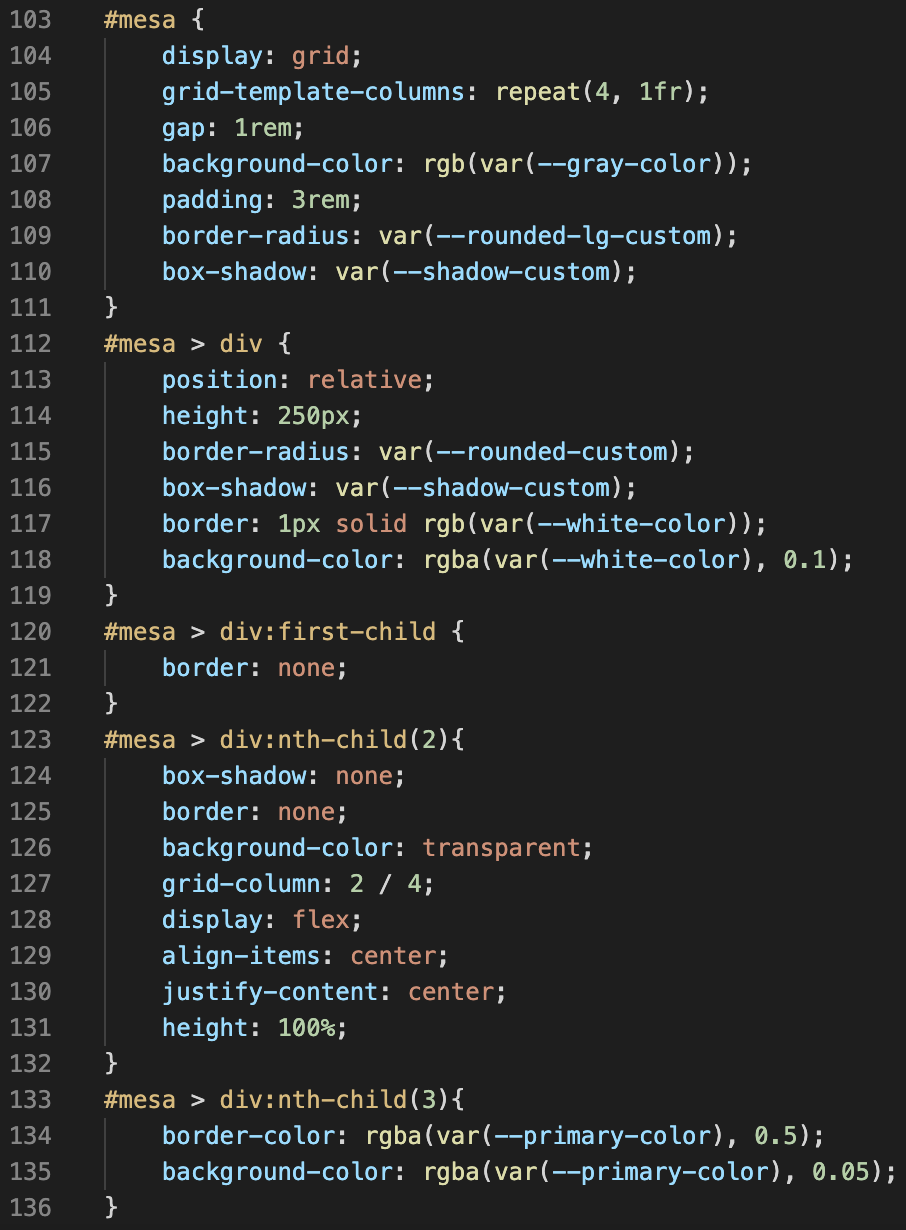
***Imagen 10.-Clase .navbar***

Especificaciones de la mesa principal el display: grid se encarga de crear las columnas para después permitir ordenar los elementos.

Se define la pseudo clase first-child donde se captura el primer div que está dentro del ID mesa.

El segundo div es donde se encuentra el botón barajar el cual es capturado por nth-child(2).

first-child permite ordenar los elementos en las columnas anteriormente creadas ordenando.



***Imagen 11.-Mesa principal***

## JAVASCRIPT

Primero en la línea 1 se definen los cuatro tipos de carta: viu, cua, hex, cir.



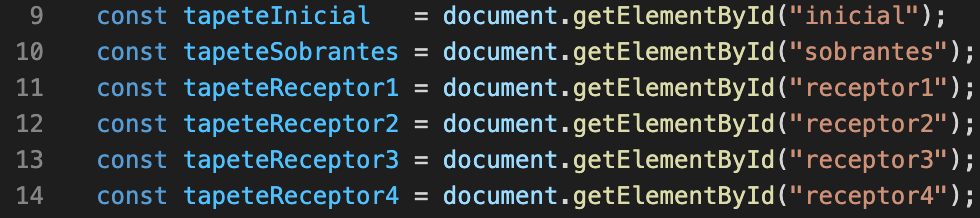
***Imagen 12.-Definición del tipo de carta***

Luego se define la lista de número que se utilizaran.



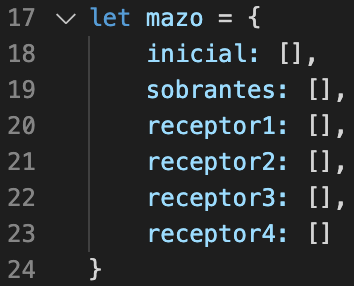
***Imagen 13.-Definición de números***

Luego se define los tapetes que se capturan mediante el ID que se definió en HTML: Inicial, Sobrantes, Receptor 1, Receptor 2, Receptor 3 y Receptor 4.



***Imagen 14.-Definición de tapetes***

Se crea un objeto que contiene un array por cada mazo: Inicial, Sobrantes, Receptor 1, Receptor 2, Receptor 3 y Receptor 4.



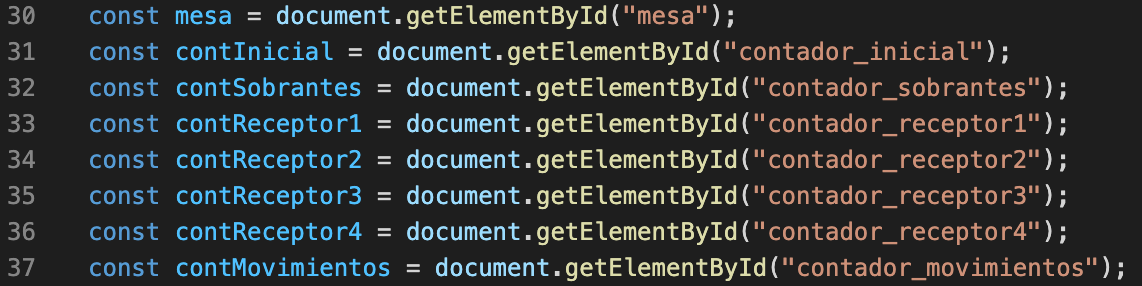
***Imagen 15.-Objeto mazo***

Luego se captura el body mediante la etiqueta body.



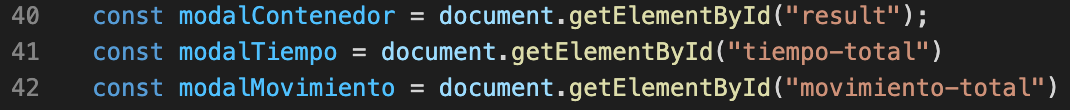
***Imagen 16.-Captura body***

Se definen los contadores de cartas utilizando el getElementById estos contadores van a definir cuántas cartas hay por cada tapete.



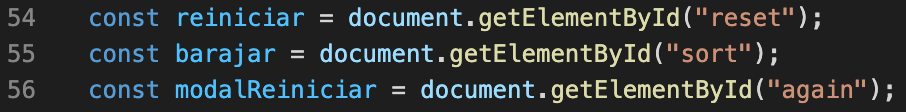
***Imagen 17.-Definición de contadores***

Luego se realiza definiendo el modal con los resultados obtenidos, ventana emergente que aparece cuando se termina el juego, aquí se captura el resultado, el tiempo y el número de movimientos.



***Imagen 18.-Captura de resultados***

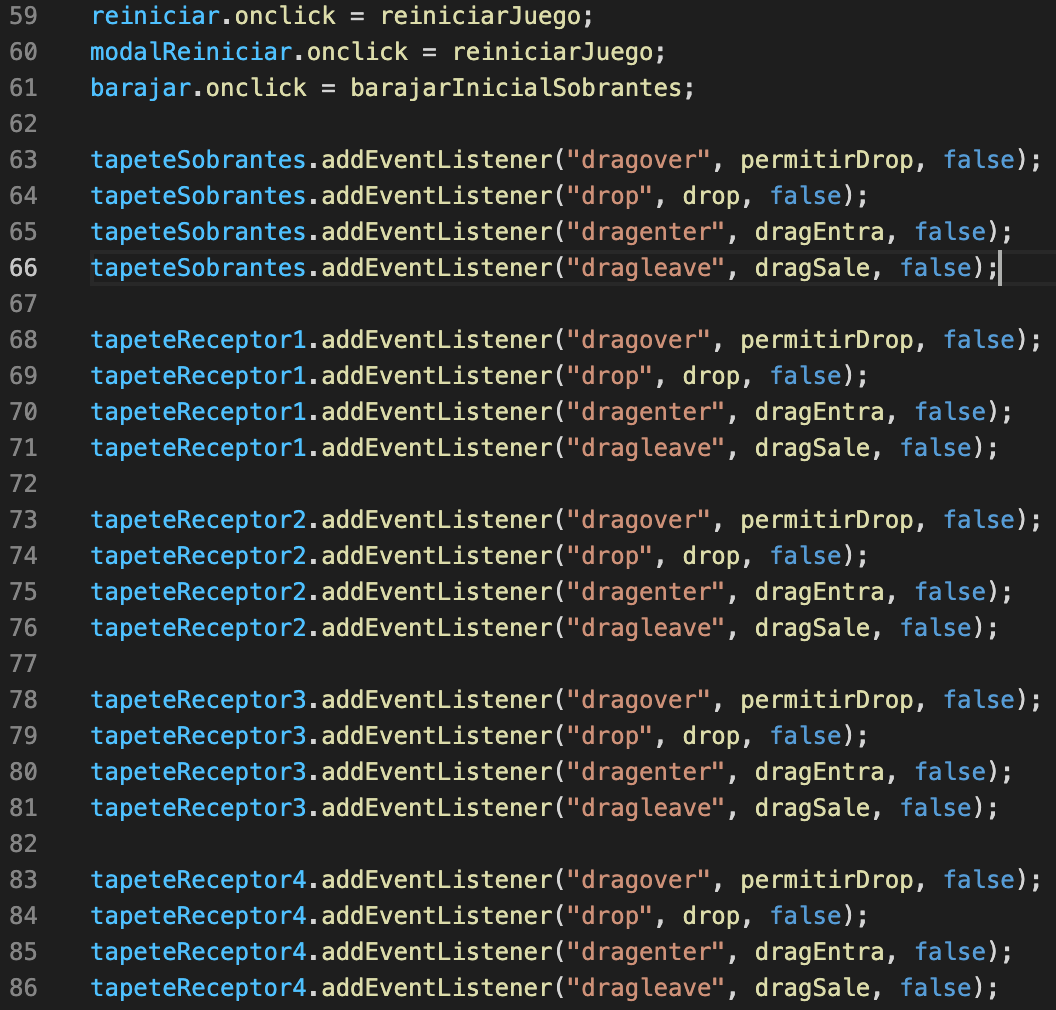
Luego se declaran las constantes reiniciar, barajar y reiniciar, que serán los botones que le permitirán al usuario interactuar con la aplicación.



***Imagen 19.-Declaración de constantes***

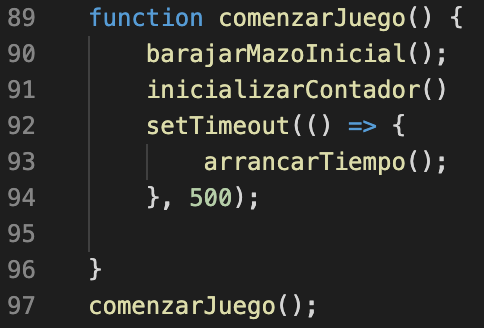
Posteriormente se procede a la asignación de las funciones le asignó que al hacer click en este botón ejecute la función reiniciar juego.

addEventListener: asigna una escucha activa de dragover, drop, dragenter y dragleave, es decir activa estas funciones cuando se arrastra una carta.



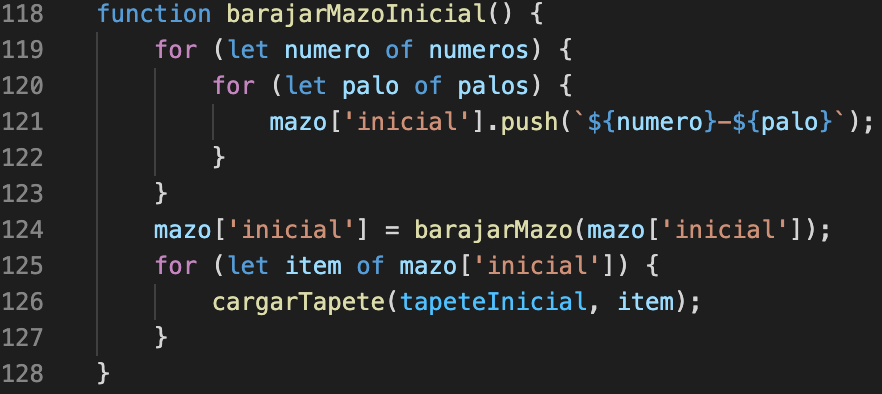
***Imagen 20.-Funciones dragover, drop, dragenter y dragleave.***

Con la función comenzarJuego() se inicia el juego, está es la función principal permite al usuario arrancar el juego desde cero.



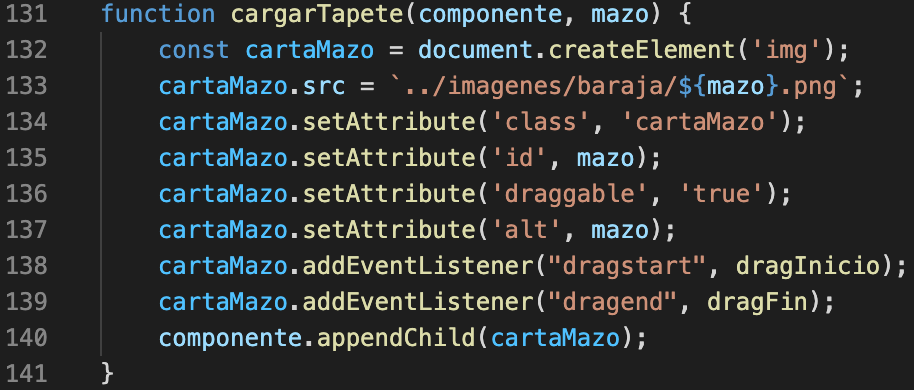
***Imagen 21.-Función comenzarJuego()***

La función barajarMazoInicialicial() con los valores de las cartas y números, es decir realiza una iteración por cada número.



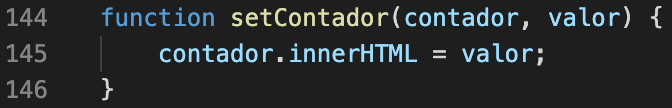
***Imagen 22.-Función barajarMazoInicial()***

Luego se crean elementos en el Modelo de Objetos del Documento DOM, con las imágenes según el array.



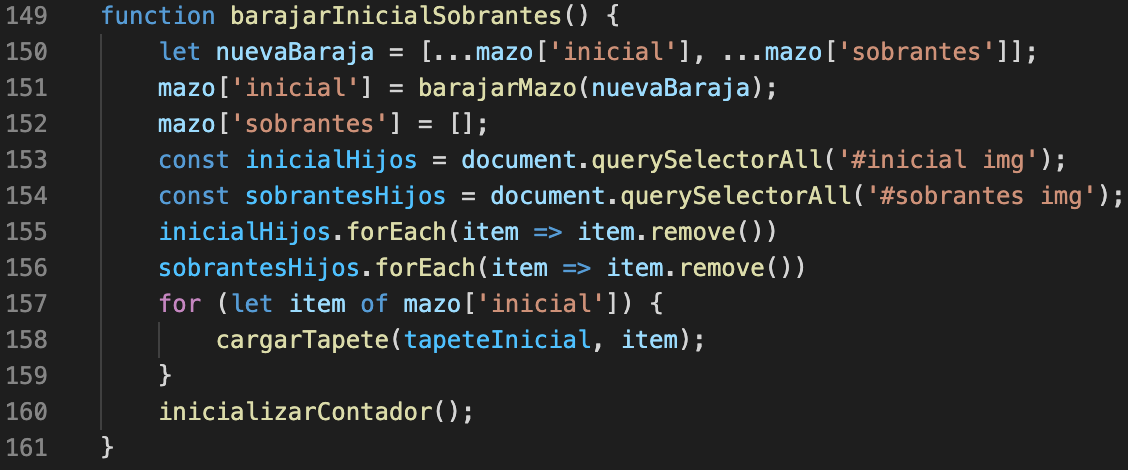
***Imagen 23.-Función cargarTapete()***

Se crea la función para modificar el contador.



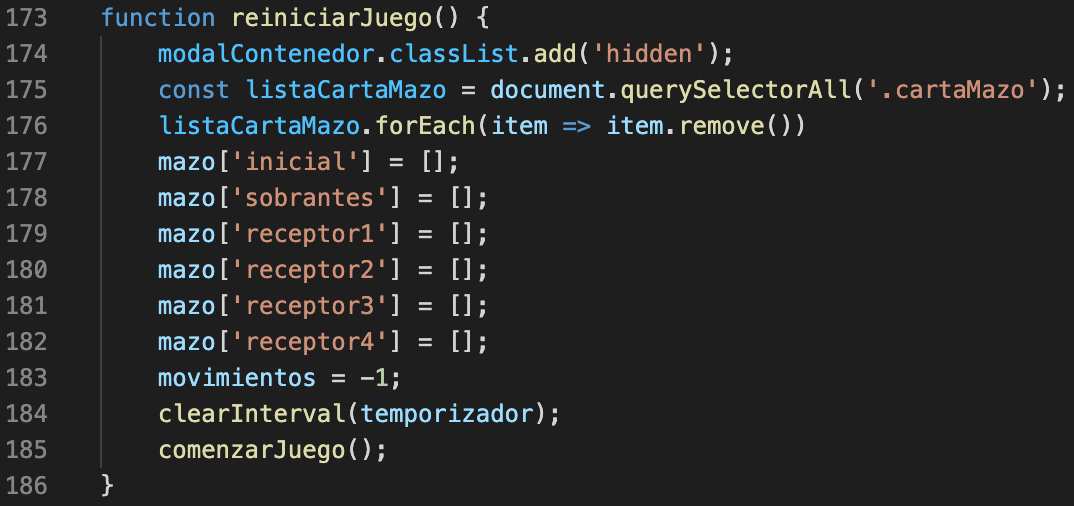
***Imagen 24.-Función setContador()***

Luego se barajan todas las cartas del Tapete Inicial y del Tapete Sobrantes.



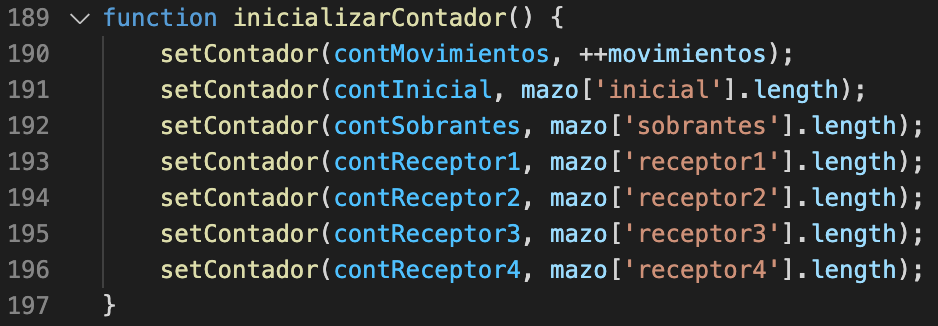
***Imagen 25.-Función barajarIniciarSobrantes()***

La siguiente función reiniciarJuego(), reinicia el juego, resetea el tiempo, limpia los mazos, oculta el modal, es decir deja los valores en su estado original.



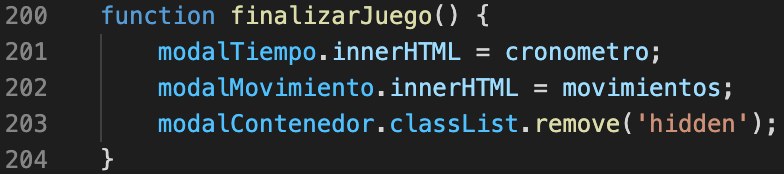
***Imagen 26.-Función reiniciarJuego()***

La siguiente función inicializarContador() actualiza el contador de cartas por tapete y el contador de movimientos.



***Imagen 27.-Función inicializarContador()***

Esta función finalizarJuego() le muestra al usuario una alerta con su información y reinicia el juego.



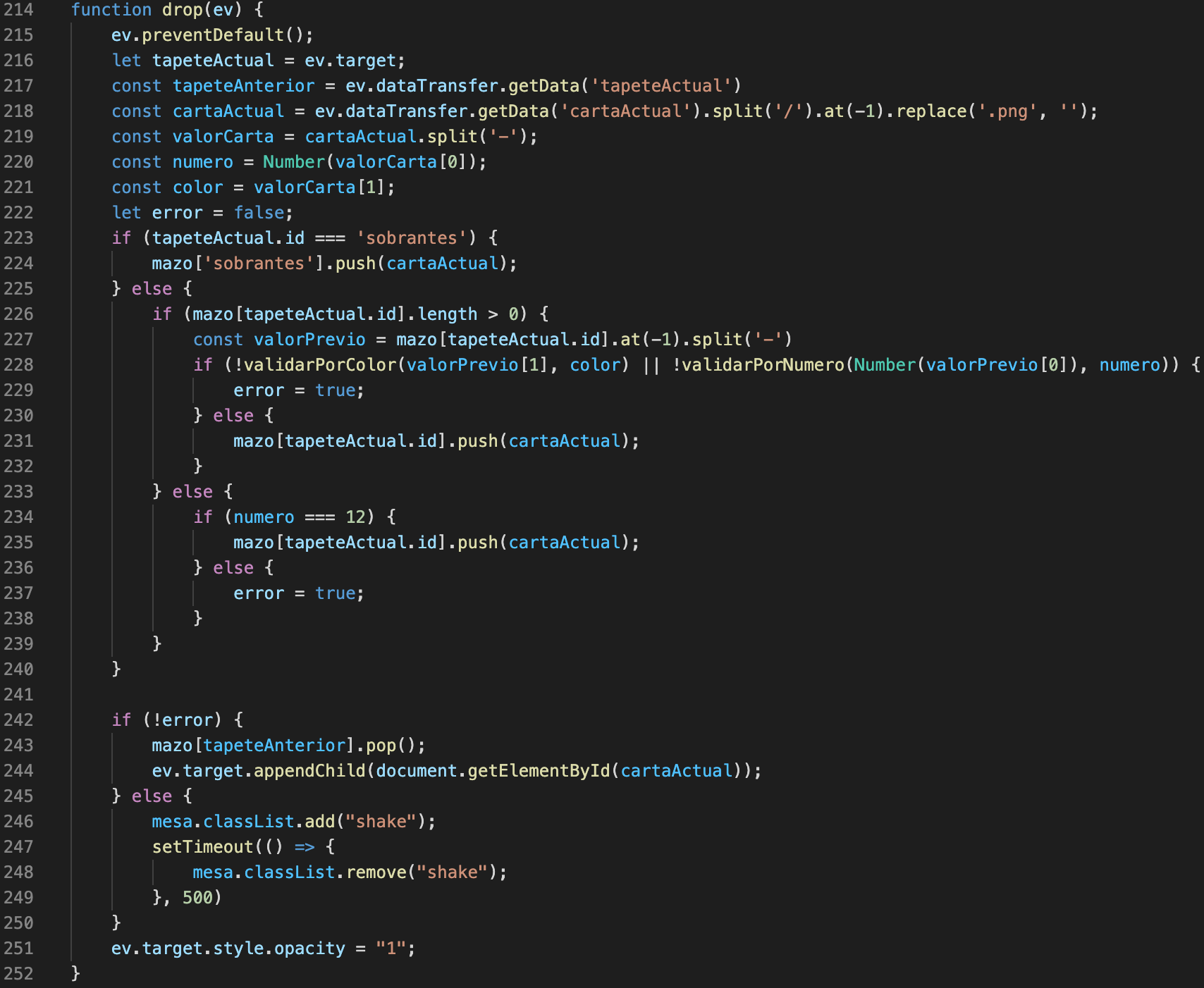
***Imagen 28.-Función finalizarJuego***

La siguiente función dragInicio() se creó con el fin de que se ejecute cuando se arrastra una carta, envíe el ID de la carta y el padre contenedor.



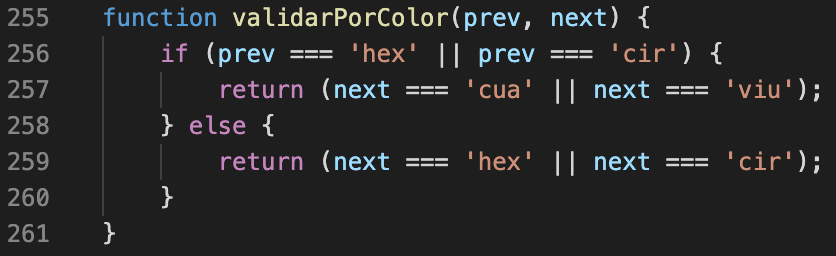
***Imagen 29.-Función dragInicio()***

Posteriormente se crea la función drop() para ser ejecutada cuando se suelte una carta dentro de otro tapete contenedor, realice la validación.



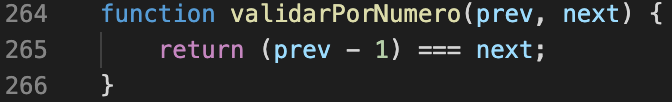
***Imagen 30.-Función drop()***

La siguiente función validarPorColor() realiza la validación por color de cada una de las cartas.



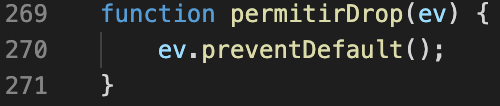
***Imagen 31.-Función validarPorColor()***

En esta función validarPorNumero() se valida cada una de las cartas por su número.



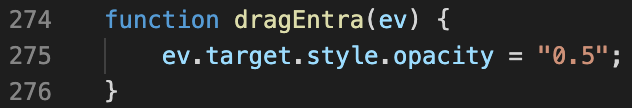
***Imagen 32.-Función validarPorNumero()***

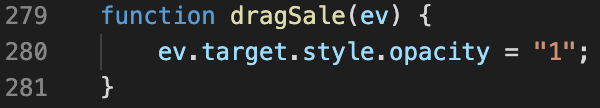
Definiendo esta función permitirDrop() le permite al usuario arrastrar cada una de las cartas.



***Imagen 33.-Función permitirDrop()***

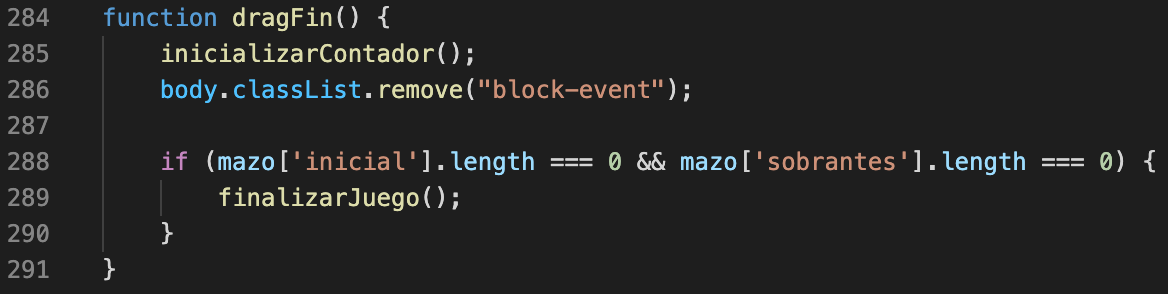
Las siguientes funciones dragEntrar() y dragSale() generan el efecto de opacidad cuando la carta entra o sale de un tapete.





***Imagenes 34 y 35.-Función dragEntra() y DragSale()***

Y esta última función dragFin(), se ejecuta cuando el usuario suelta la última carta.



***Imagen 36.-Función dragFin()***

## RESULTADO FINAL

El usuario cuenta con un botón de reinicio, el cual le permite empezar el juego desde cero.

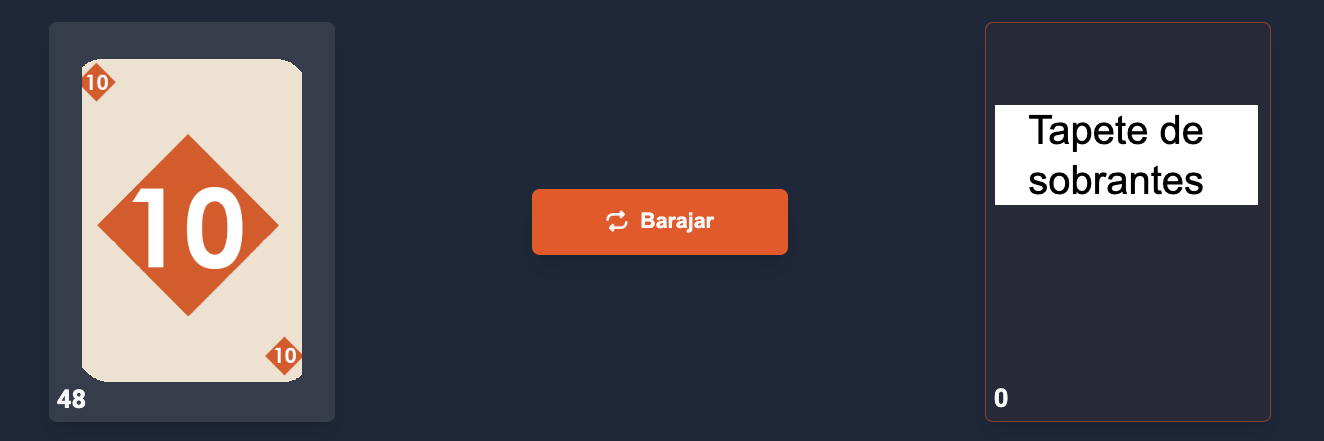
***Imagen 37.-Botón reiniciar***

Cuenta con un botón de barajar, que permite cuando el tapete inicial quede vacío, restablecerlo con las cartas que están en el tapete de sobrantes y así empezar la búsqueda de nuevo o cuando el usuario lo considere pertinente puede realizar está acción para restablecer el tapete inicial.



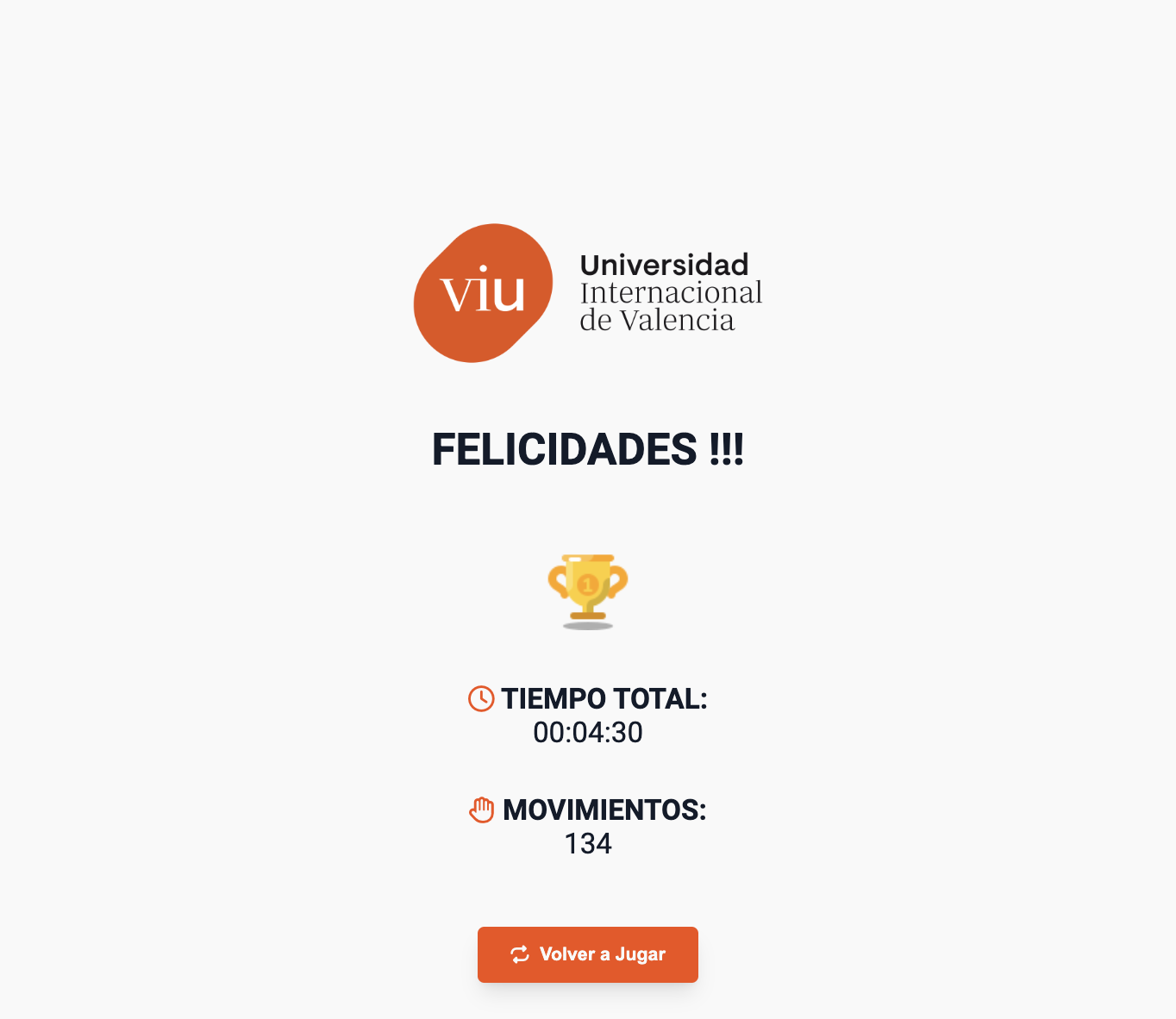
***Imagen 38.-Botón barajar***

Así mismo podrá arrastrar las cartas al tapete de sobrantes hasta que encuentre la carta indicada que pueda depositar en los tapetes 3,4,5 y 6.



***Imagen 39.-Tapete sobrantes***

Cuando arrastre la última carta al tapete correspondiente y termine el juego, el usuario visualizará un mensaje de felicitación con los datos obtenidos en el transcurso del juego, como son: el tiempo total empleado y el número de movimientos realizados para conseguir finalizarlo. En el mismo mensaje se incluye un botón volver a jugar que le permitirá volver al juego y empezar una nueva partida.



***Imagen 40.-Modal***

# CONCLUSIONES

La presente actividad consistió en un ejercicio de exploración al lenguaje de programación JavaScript y en los lenguajes de marcado HTML y lenguaje de estilos CSS, de manera que pudimos reforzar nuestros conocimientos vistos en clase y además realizar investigación para ejercitar la competencia del aprendizaje permanente.

Estamos convencidos que el presente ejercicio, es una buena práctica para poder identificar los elementos básicos de programación que deben existir en dicho paradigma, así como poner en práctica la declaración de variables y constantes.

Consideramos que la práctica fue exitosa y se cubrieron con los requisitos solicitados en la misma.