

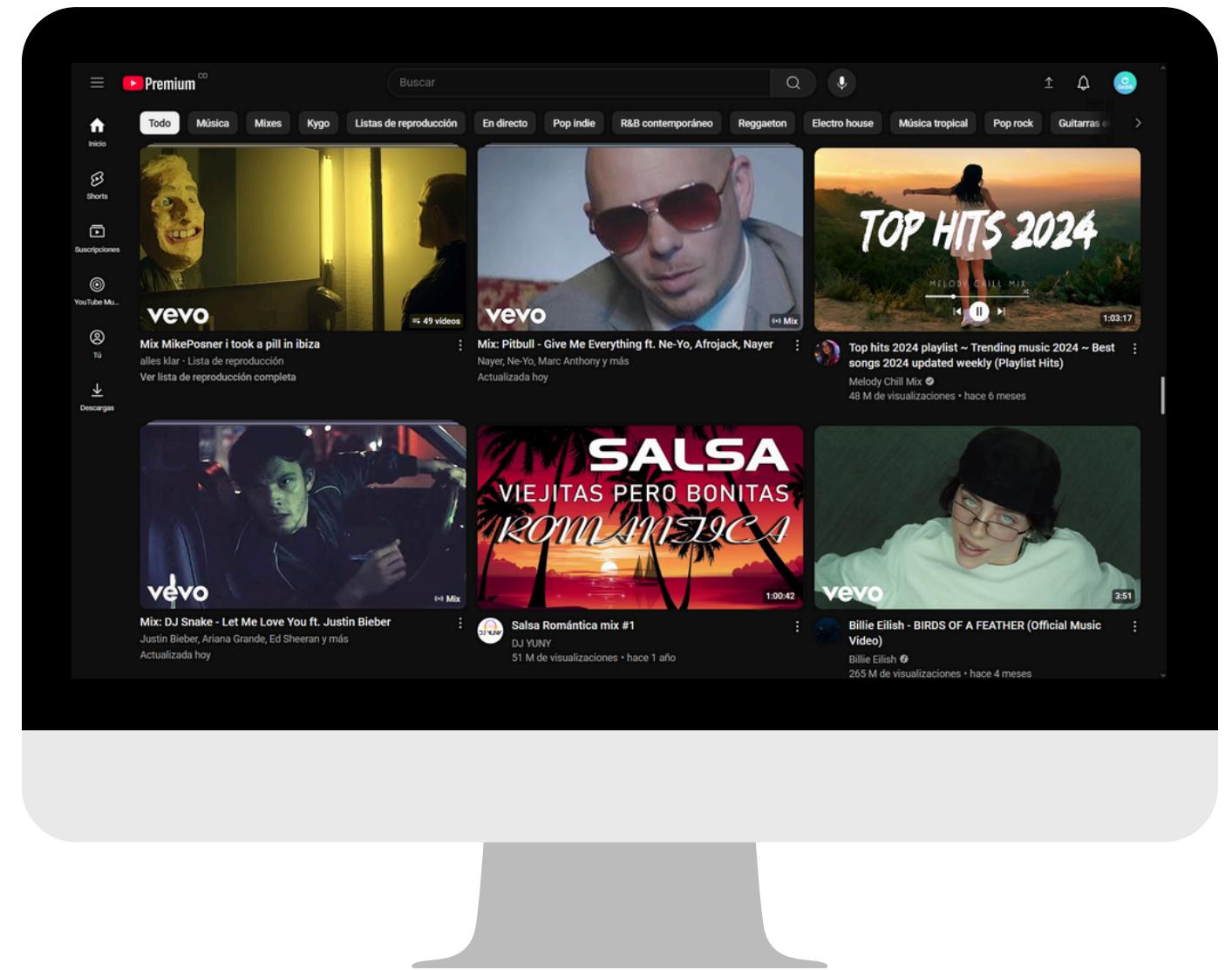
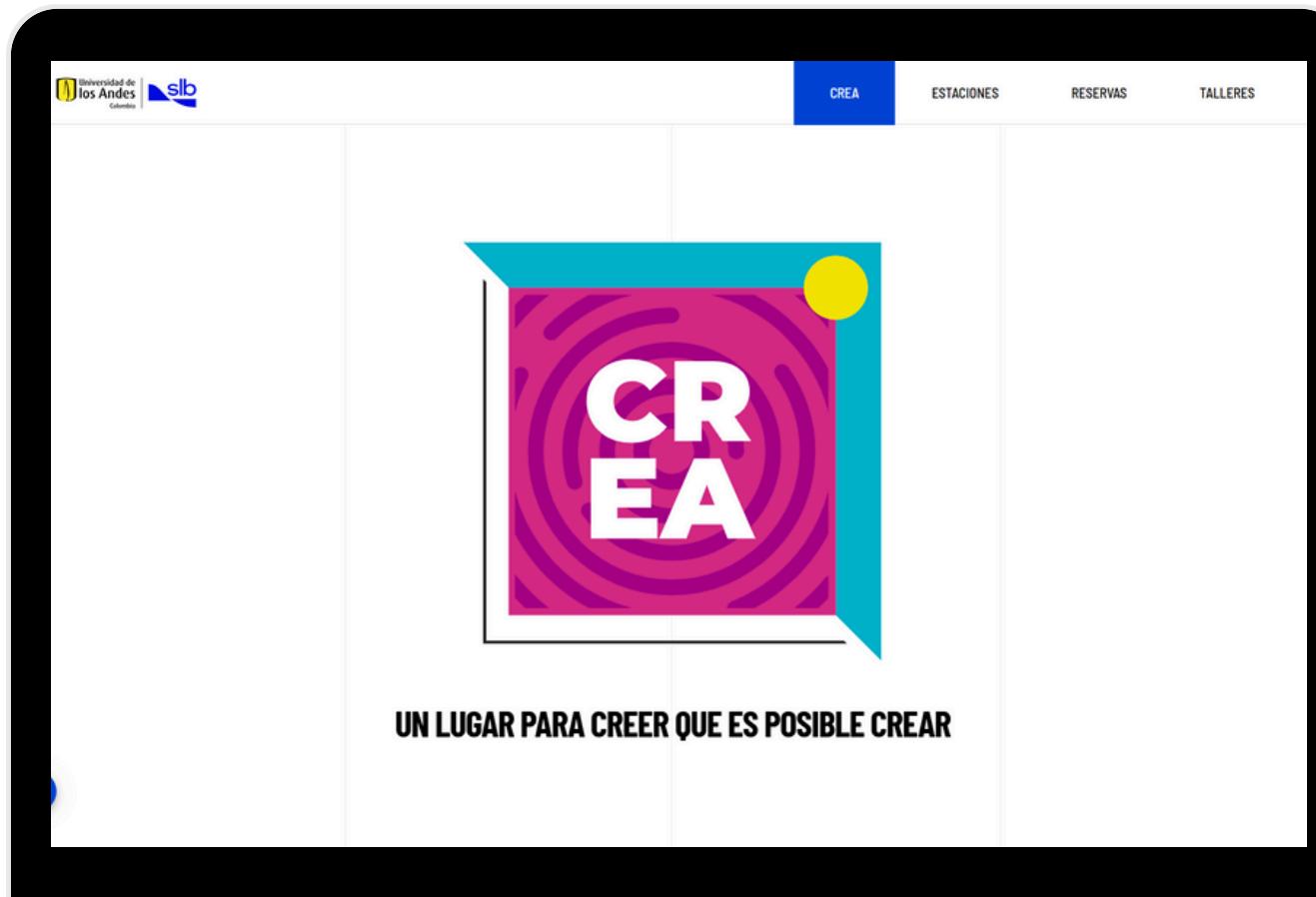


# ¡CREA tu Propio Videojuego!

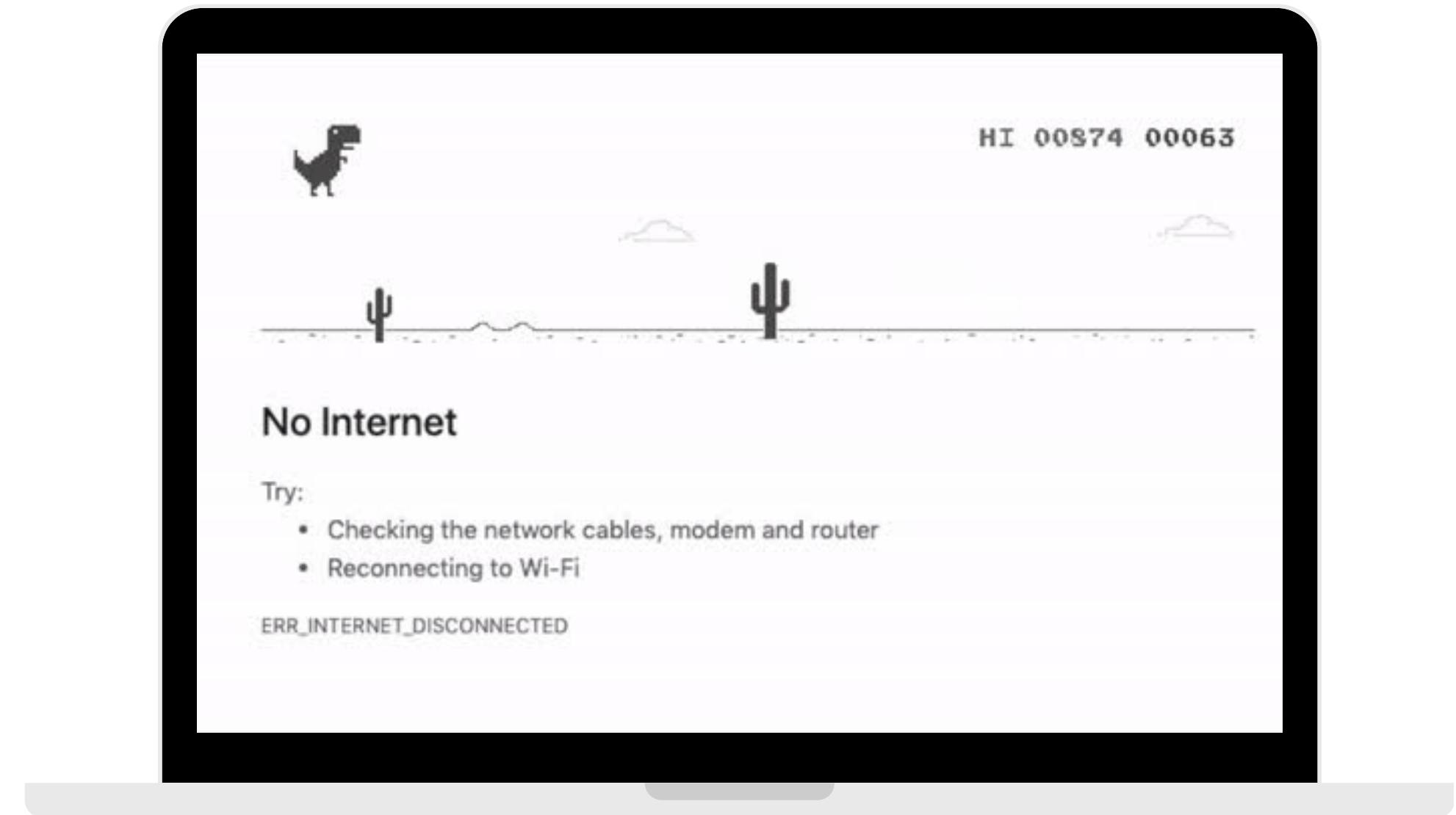
Explorando la programación web



# Programación web



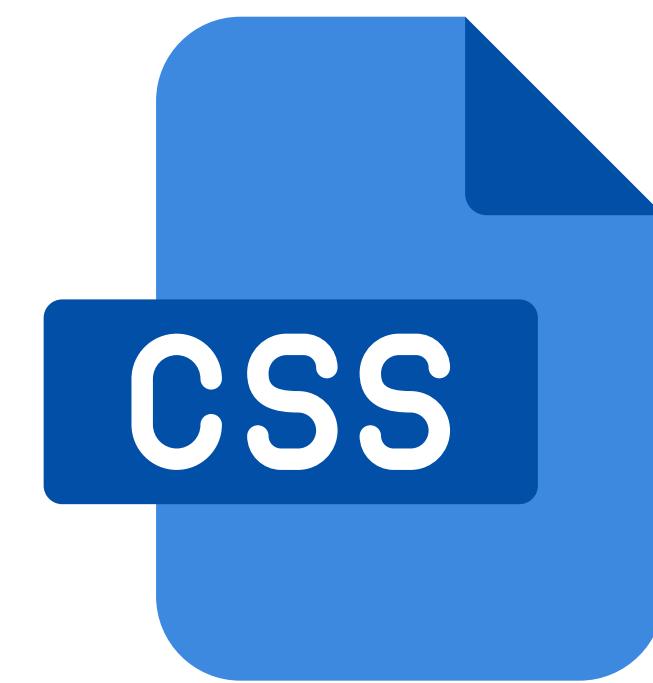
# Programación web - Juegos



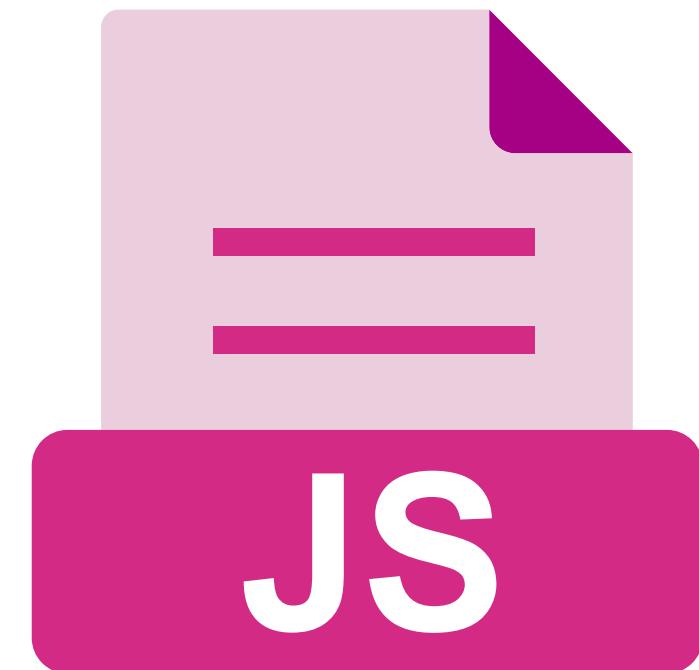
# ¿Cómo funciona una página web?



Estructura

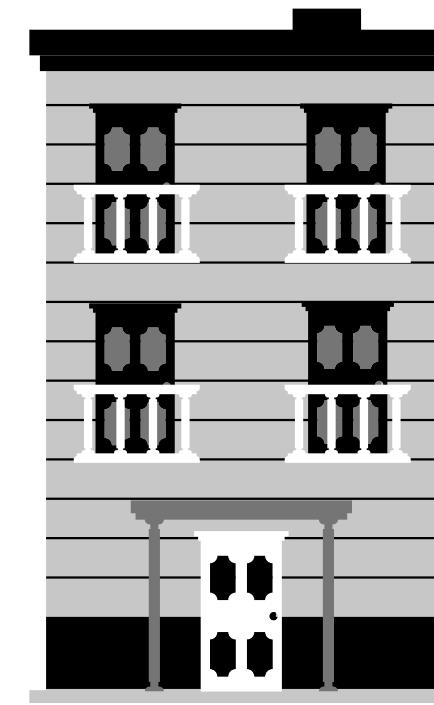


Estilo y Diseño

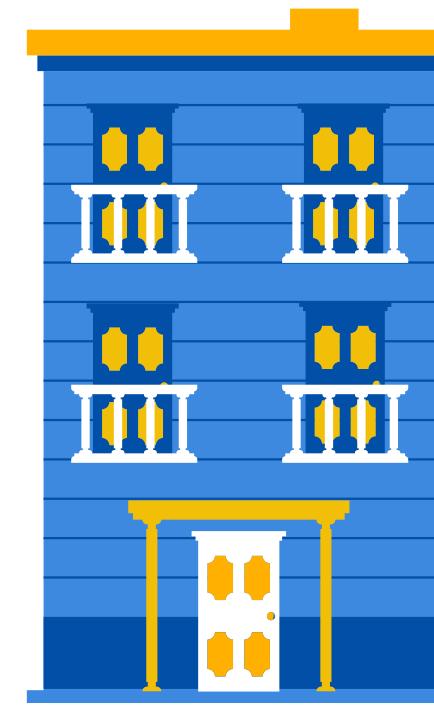


Interacción

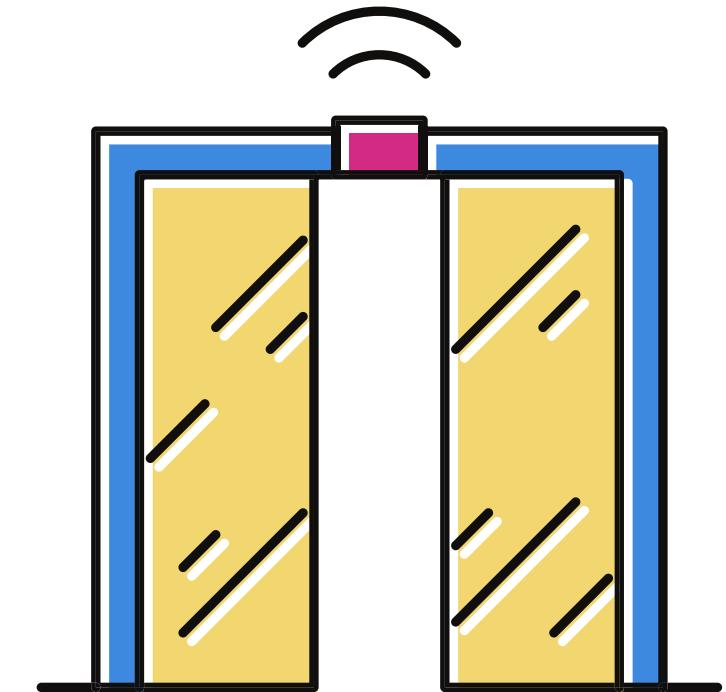
# ¿Cómo funciona una página web?



**HTML**

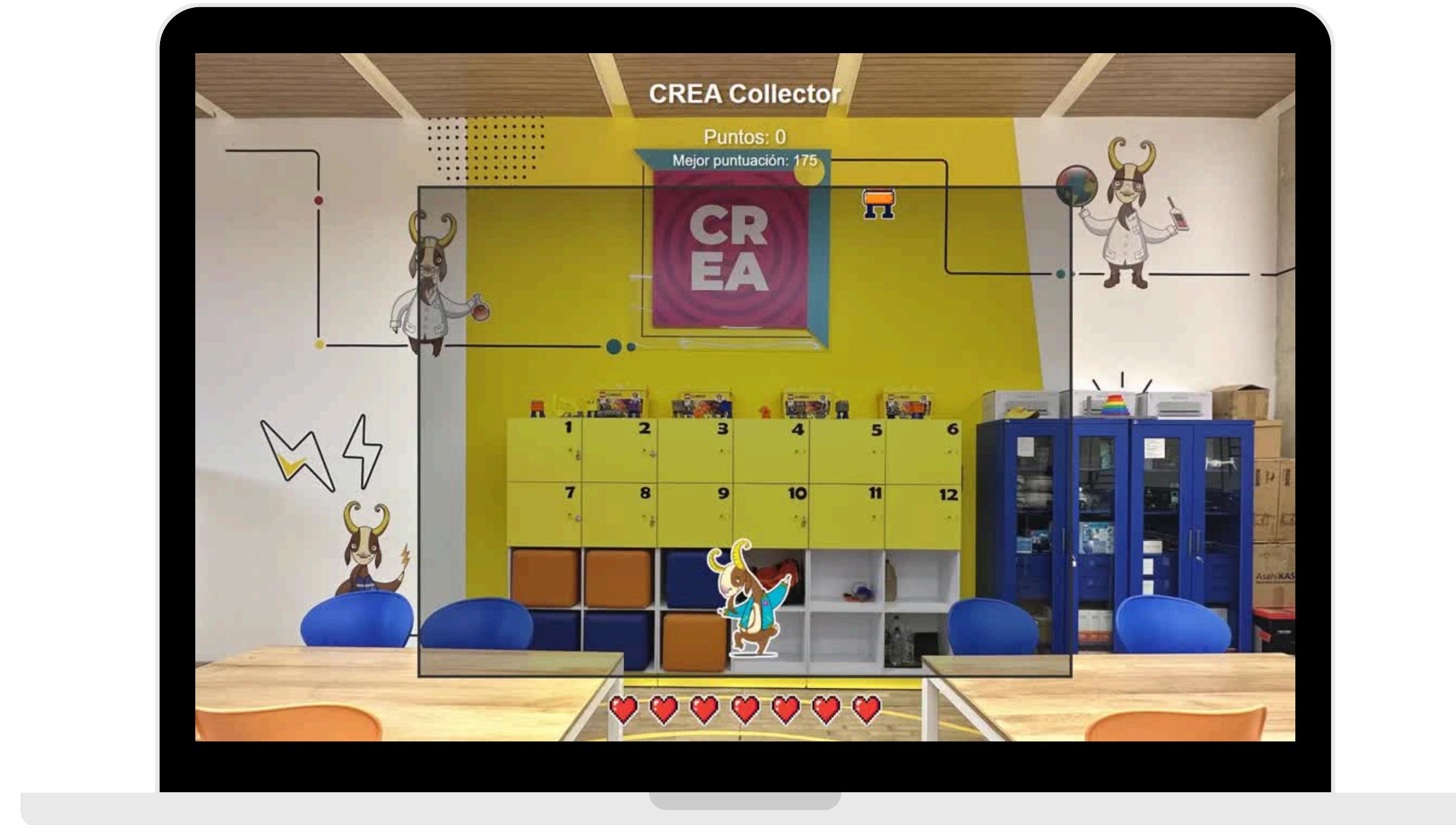


**CSS**

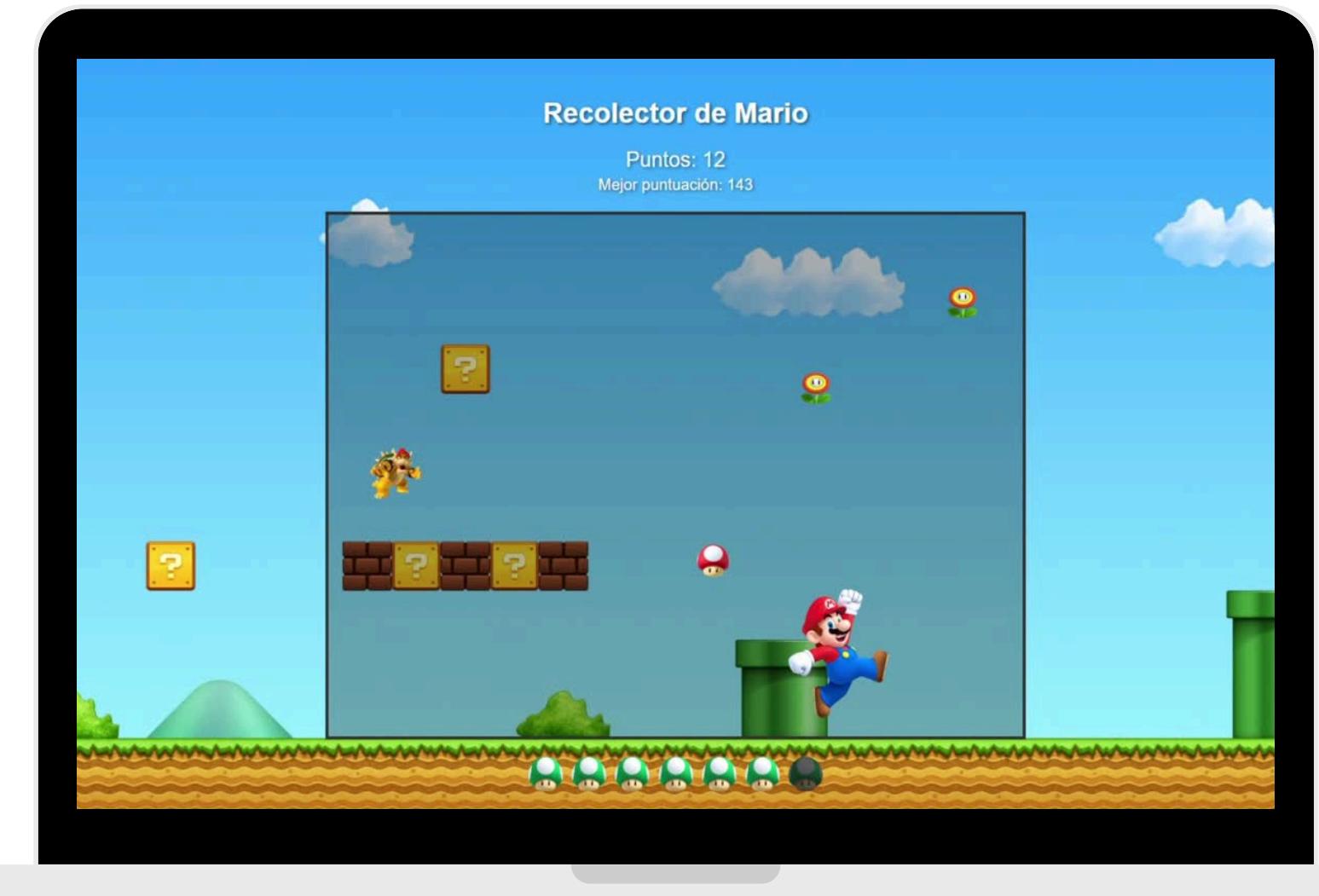
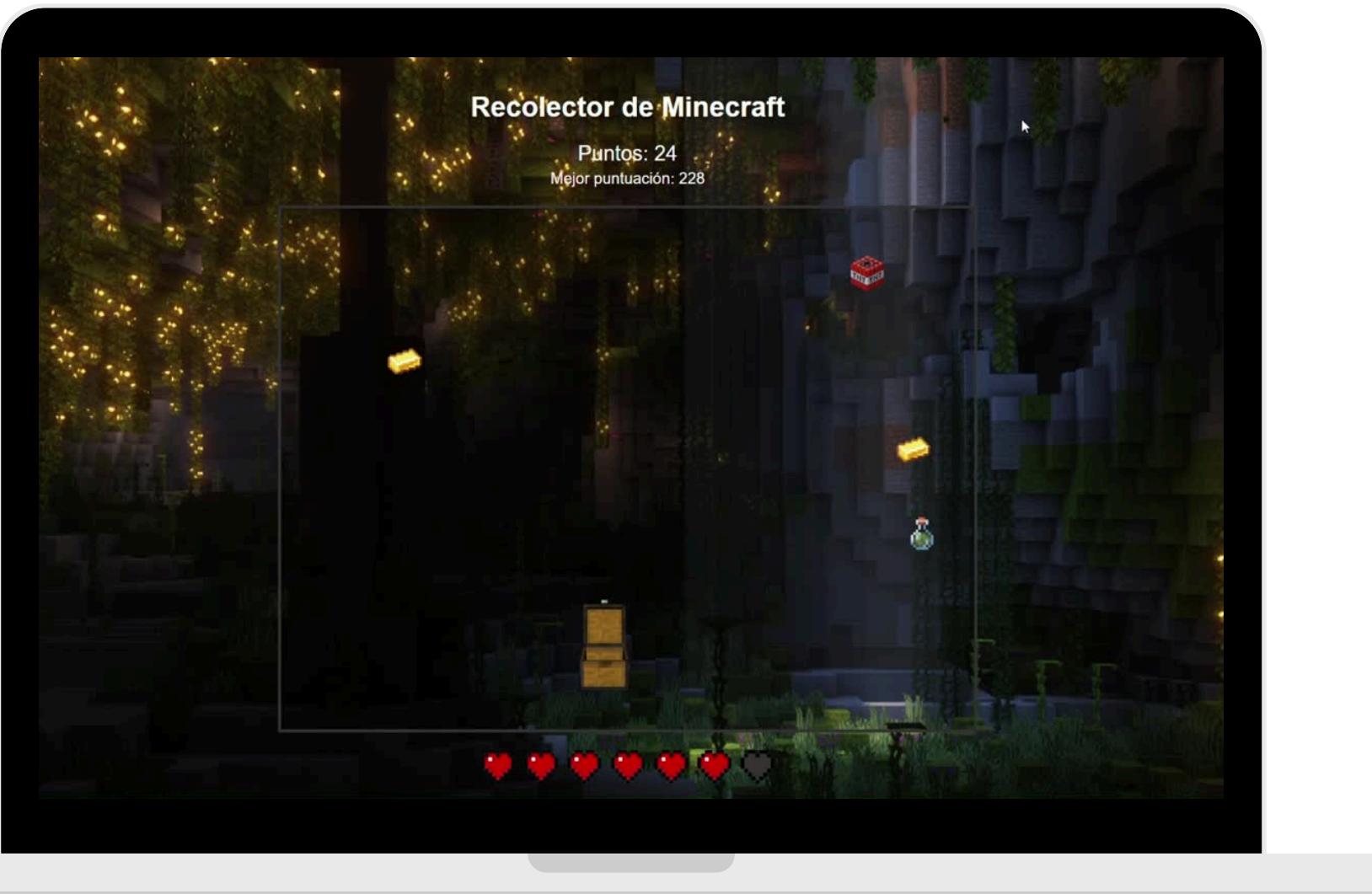


**JavaScript**

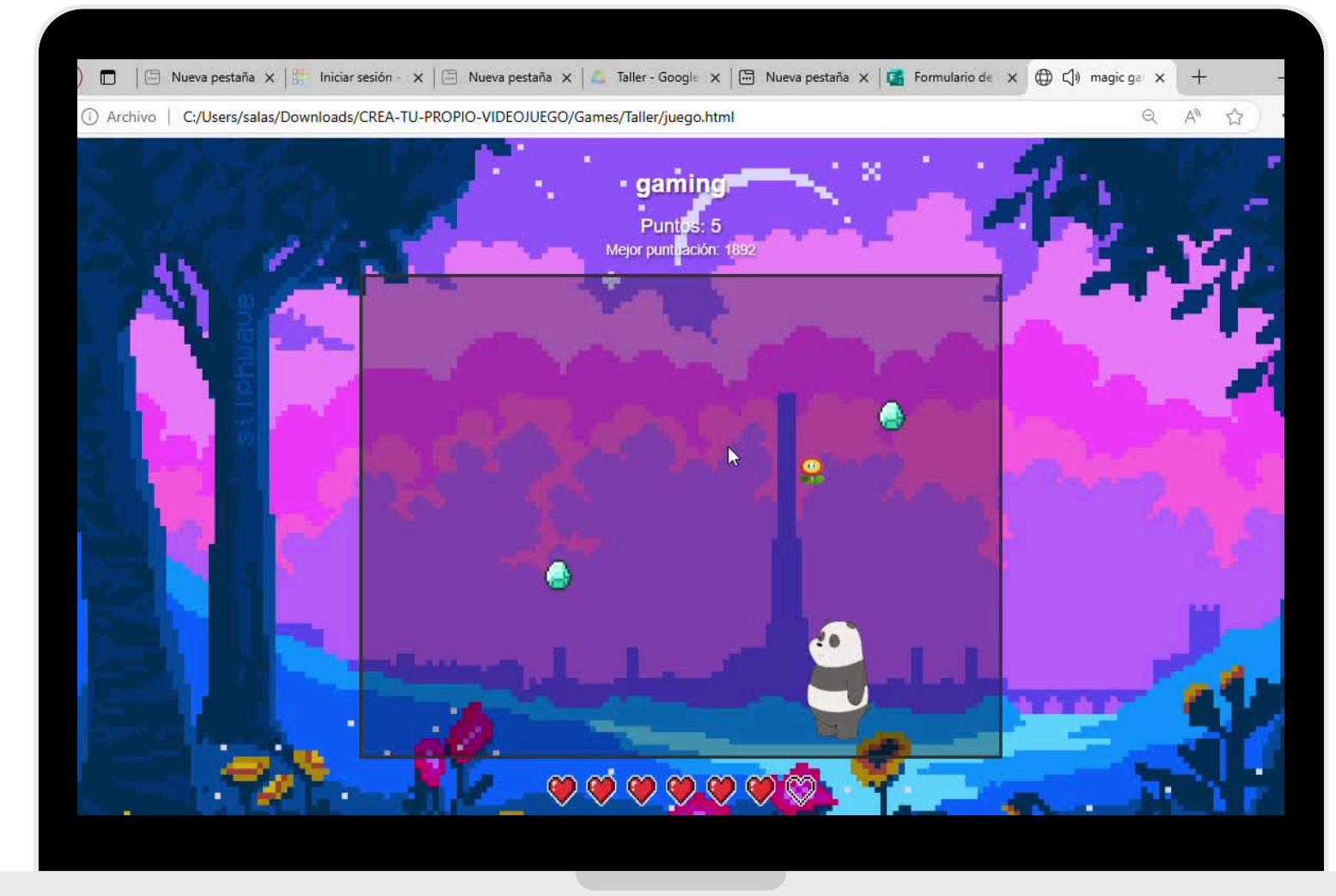
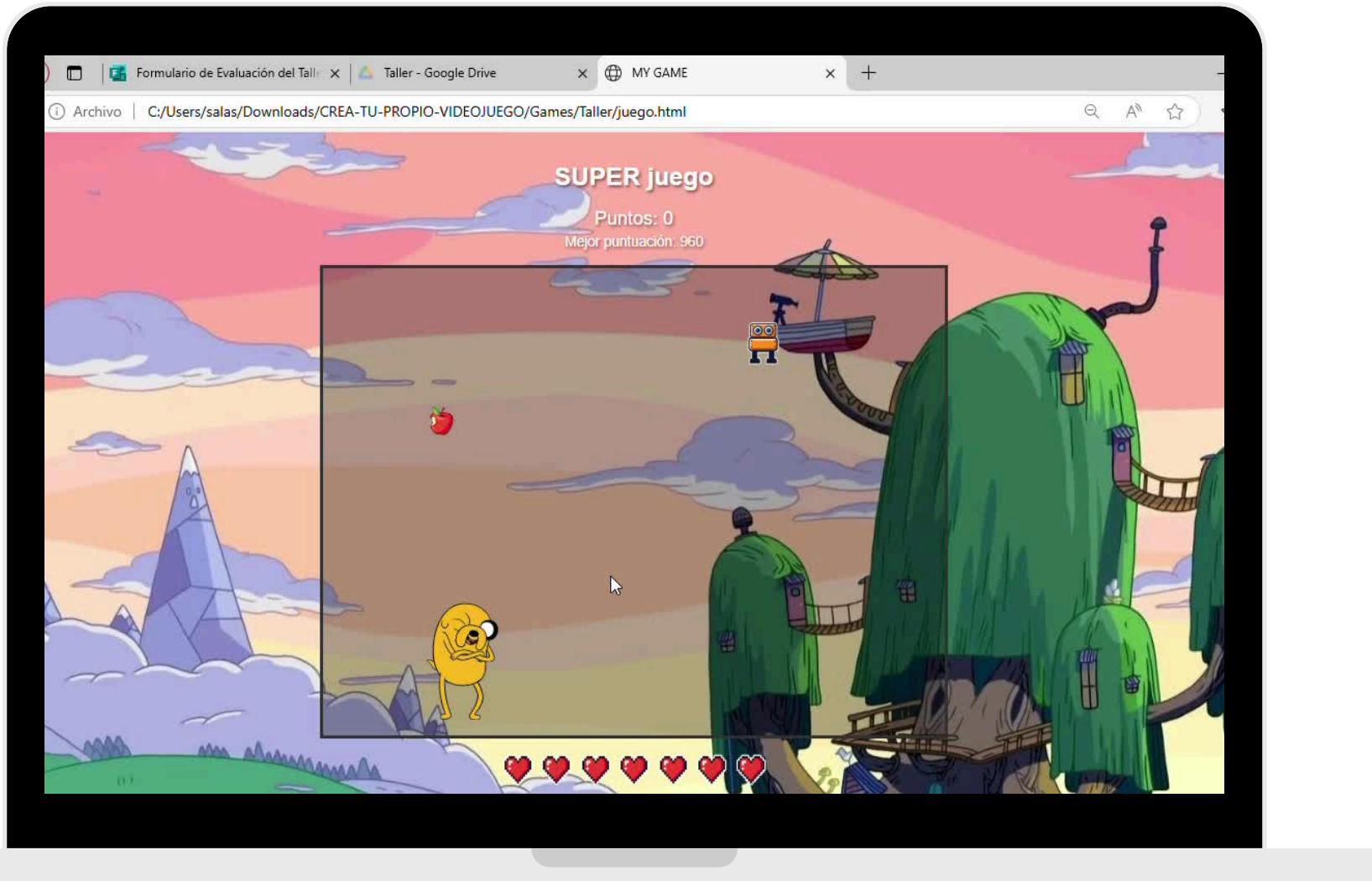
# Ejemplos de lo que vamos a crear



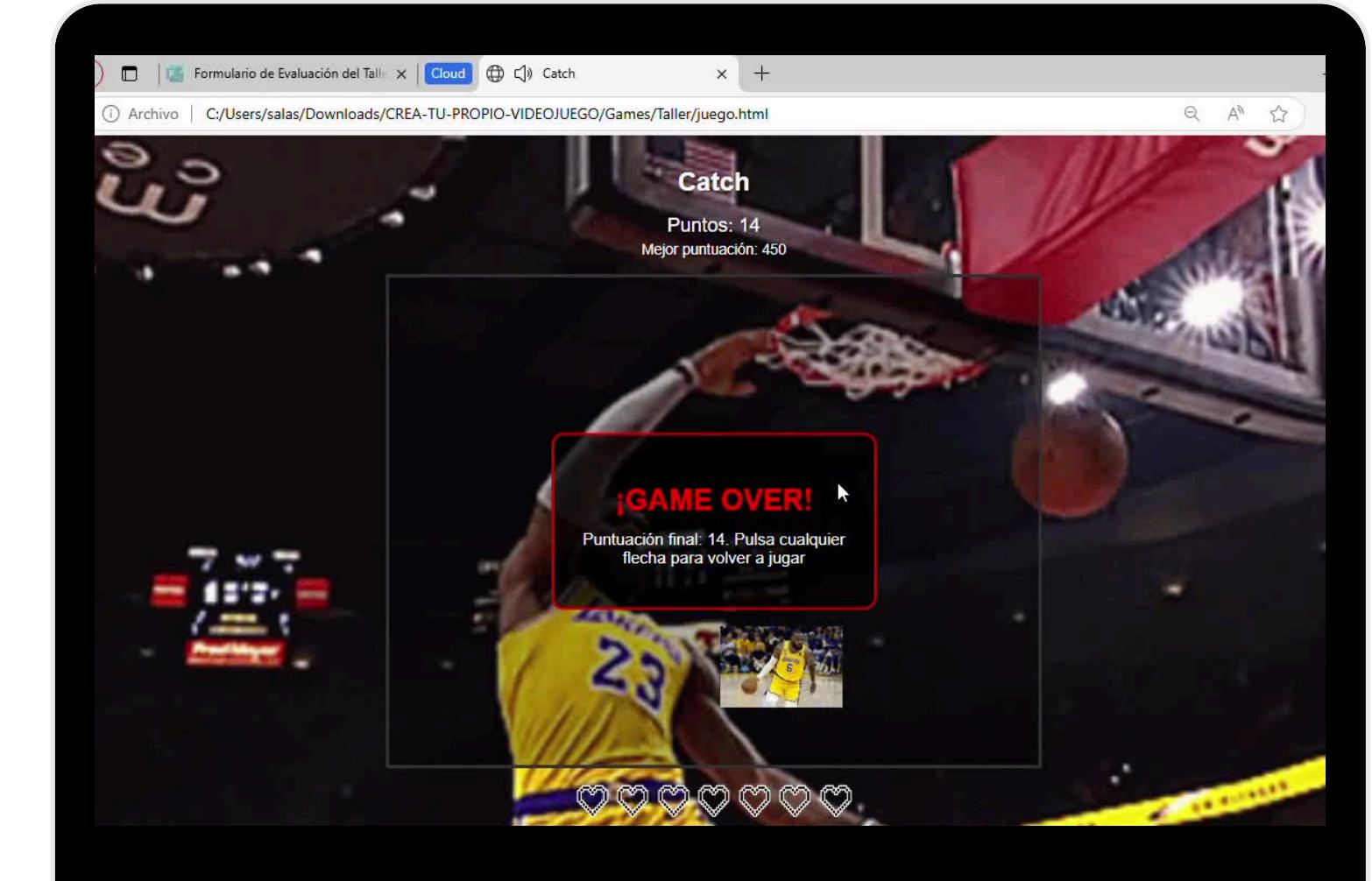
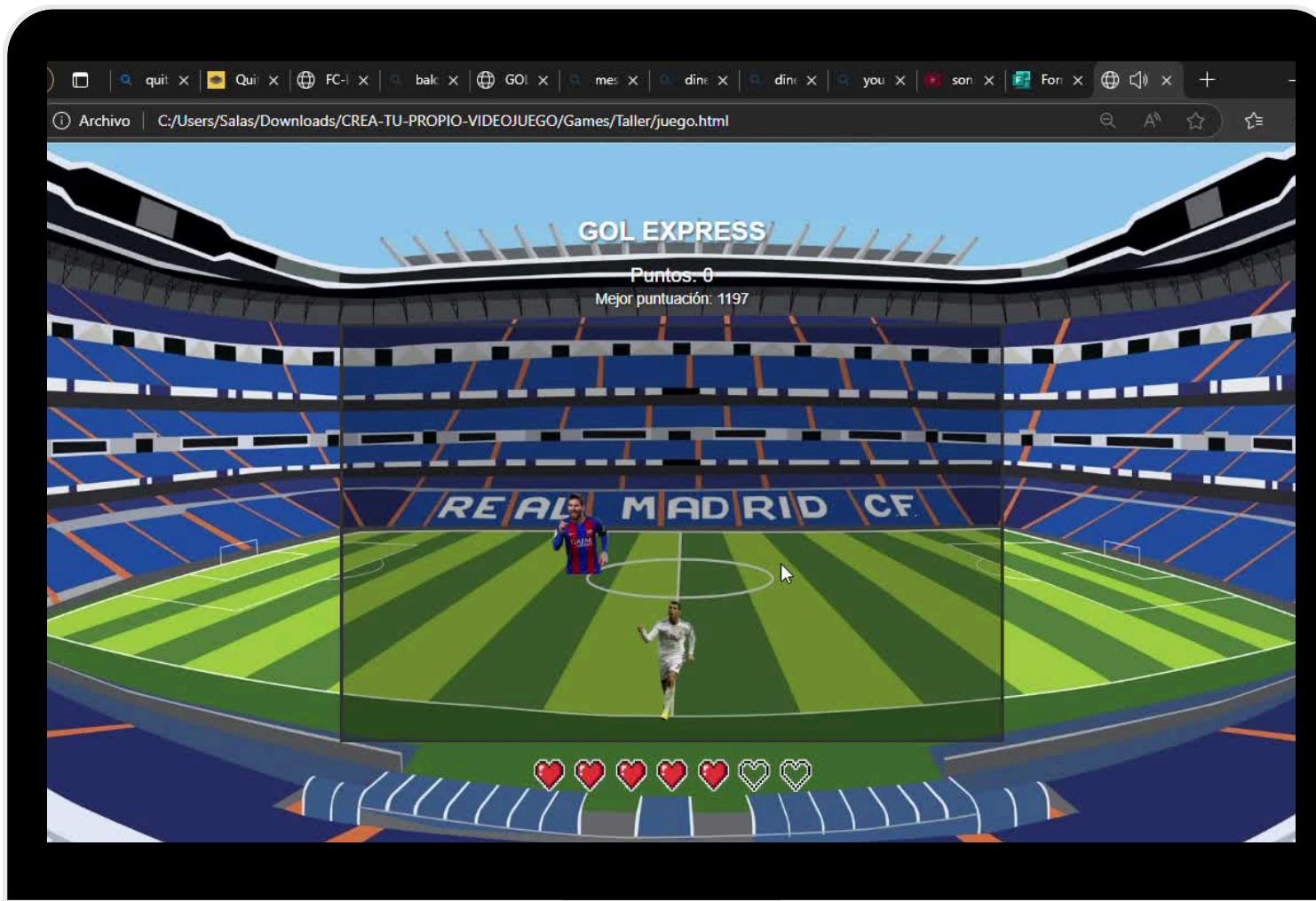
# Ejemplos de lo que vamos a crear



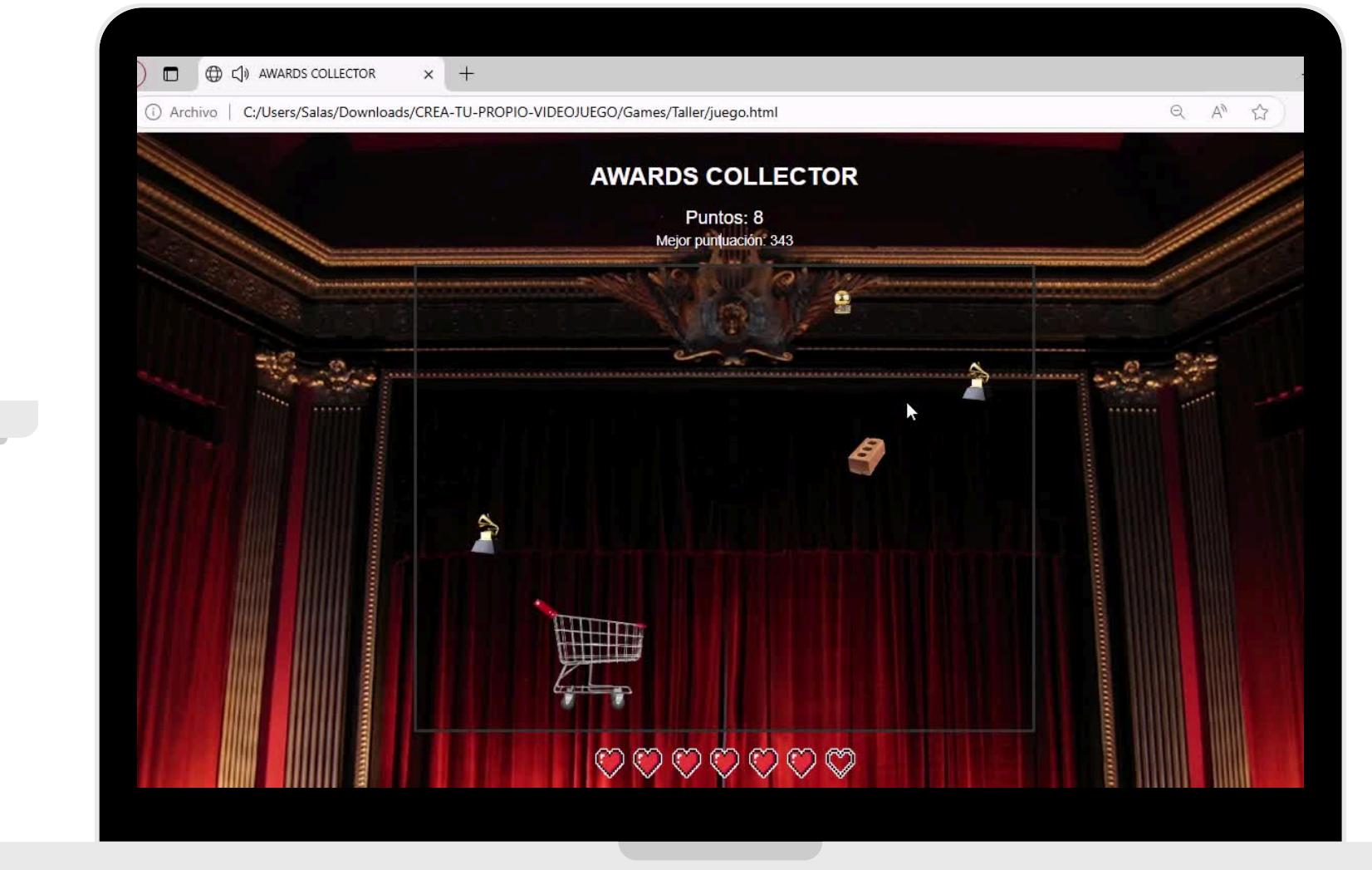
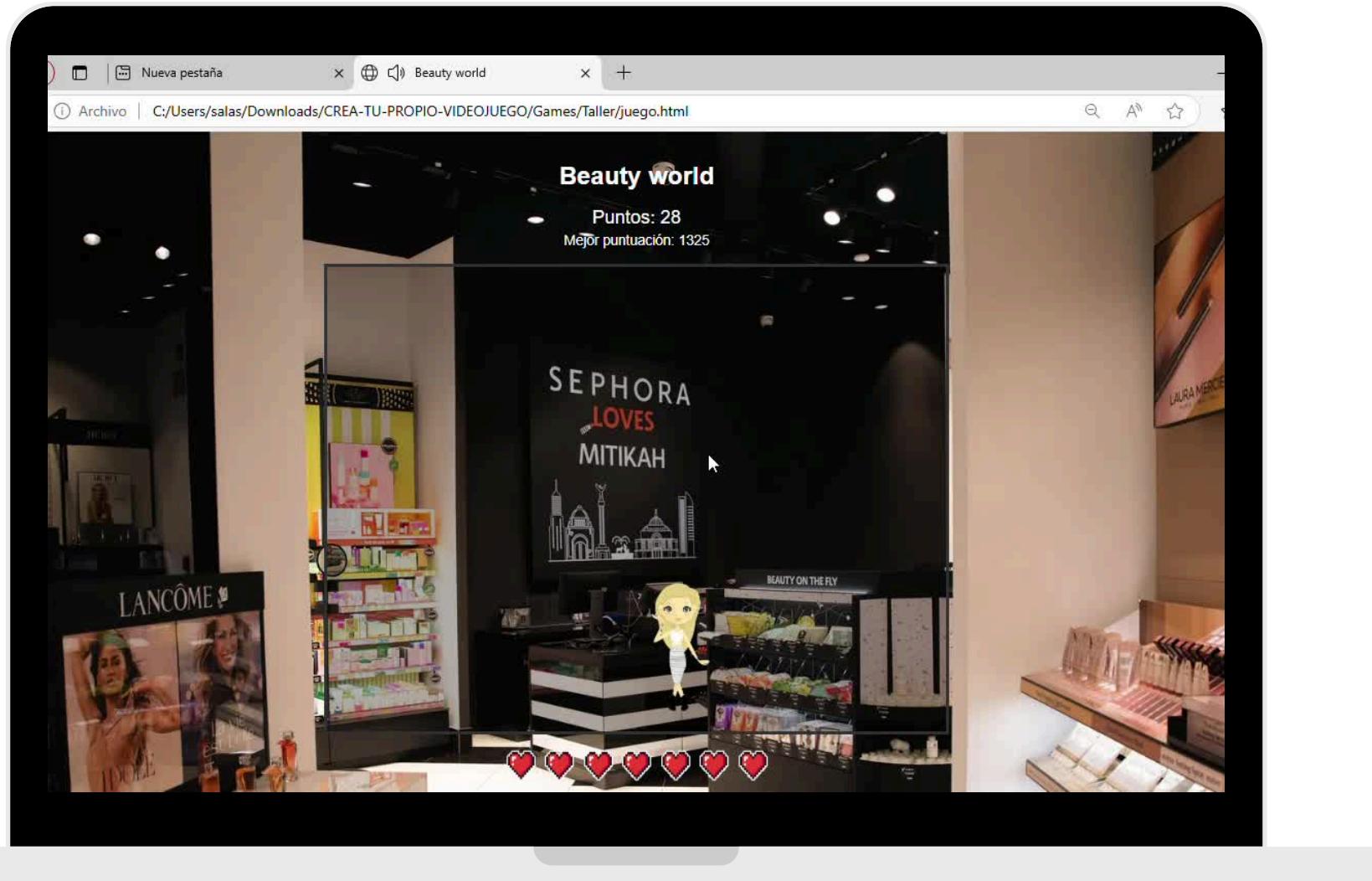
# Ejemplos de lo que vamos a crear



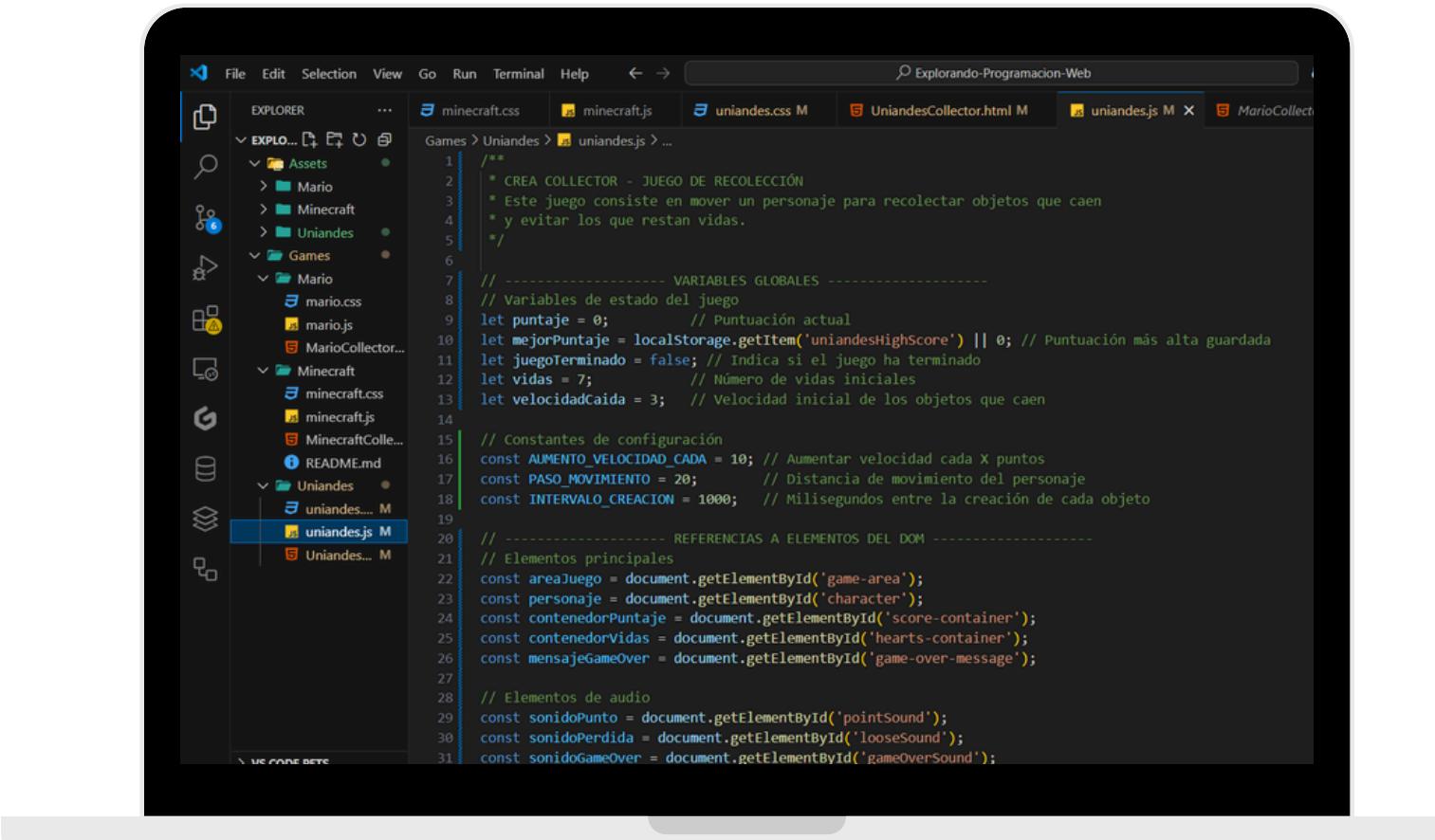
# Creaciones de estudiantes



# Creaciones de estudiantes



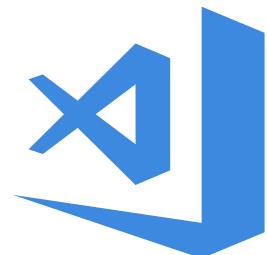
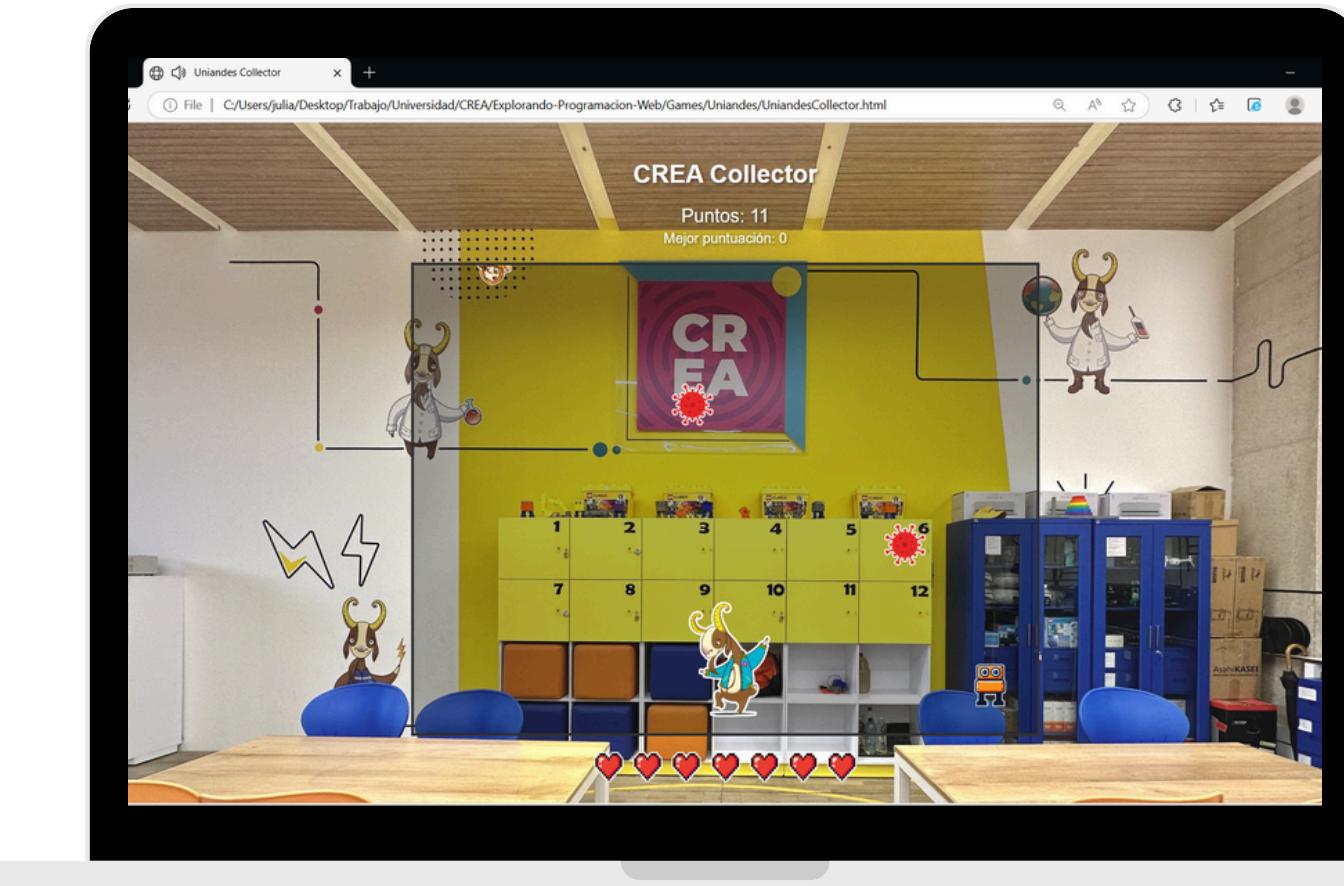
# ¿Qué vamos a utilizar?



```

1 /**
 * CREA COLLECTOR - JUEGO DE RECOLECCIÓN
 * Este juego consiste en mover un personaje para recolectar objetos que caen
 * y evitar los que restan vidas.
 */
2
3 // Variables de estado del juego
4 let puntaje = 0; // Puntuación actual
5 let mejorPuntaje = localStorage.getItem('uniandesHighScore') || 0; // Puntuación más alta guardada
6 let juegoTerminado = false; // Indica si el juego ha terminado
7 let vidas = 7; // Número de vidas iniciales
8 let velocidadCaida = 3; // Velocidad inicial de los objetos que caen
9
10 // Constantes de configuración
11 const AUMENTO_VELOCIDAD_CADA = 10; // Aumentar velocidad cada X puntos
12 const PASO_MOVIMIENTO = 20; // Distancia de movimiento del personaje
13 const INTERVALO_CREACION = 1000; // Milisegundos entre la creación de cada objeto
14
15 // Referencias a elementos del DOM
16 // Elementos principales
17 const areaJuego = document.getElementById('game-area');
18 const personaje = document.getElementById('character');
19 const contenedorPuntaje = document.getElementById('score-container');
20 const contenedorVidas = document.getElementById('hearts-container');
21 const mensajeGameOver = document.getElementById('game-over-message');
22
23 // Elementos de audio
24 const sonidoPunto = document.getElementById('pointSound');
25 const sonidoPerdida = document.getElementById('looseSound');
26 const sonidoGameOver = document.getElementById('gameOverSound');
27
28
29
30
31

```



**Entorno de Desarrollo Integrado  
IDE**



**Explorador**



# Para iniciar

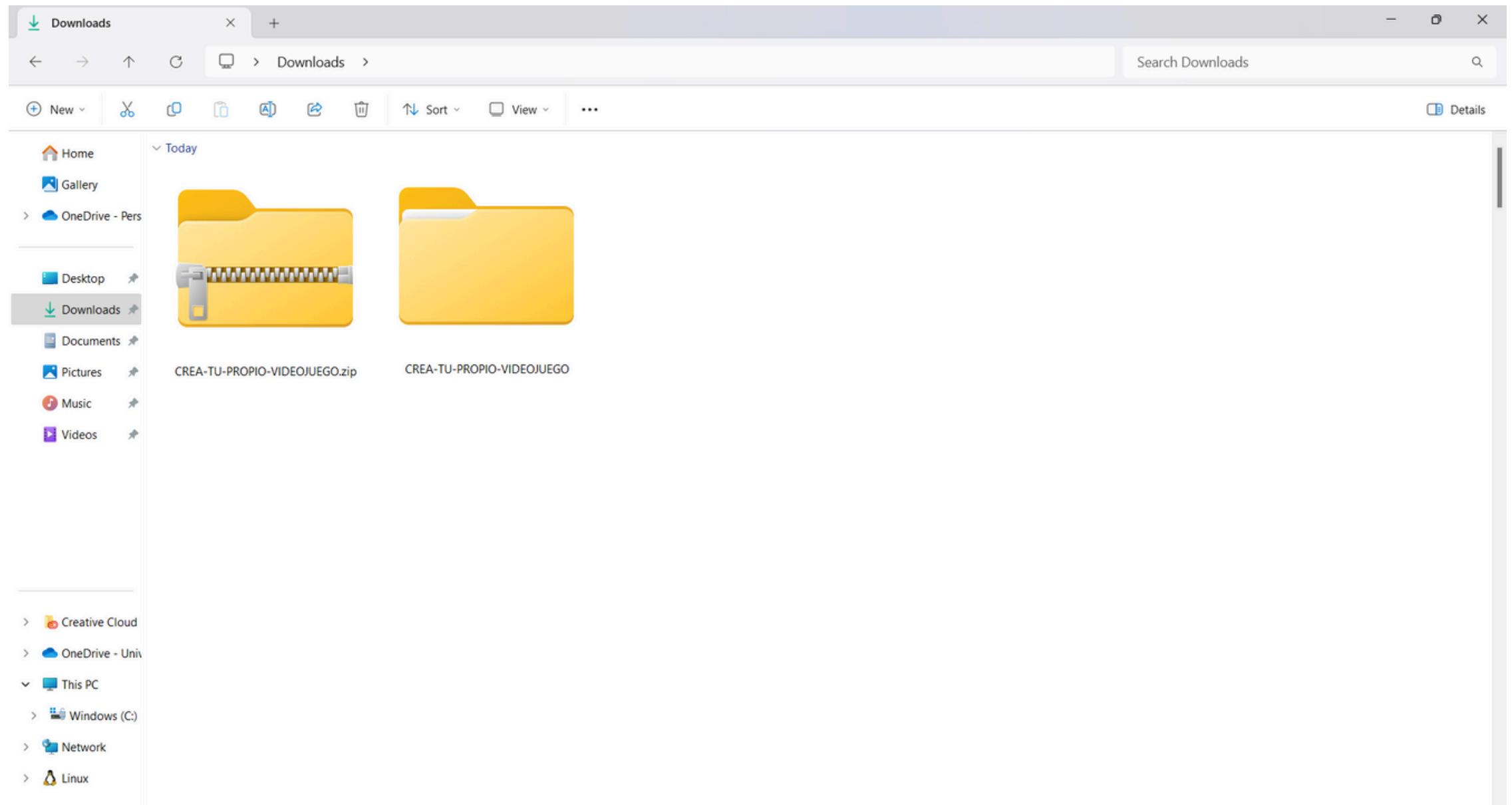
A screenshot of a Google Drive interface. The left sidebar shows navigation options like 'Nuevo', 'Página principal', 'Mi unidad', 'Ordenadores', etc. The main area shows a folder named 'Taller'. Inside, there is a single item: a zip file named 'CREA-TU-PROPIO-VIDEOJUEGO.zip'. The file is owned by 'yo' and was modified on '9 mar yo'. The size is listed as '154,2 MB'. The interface includes standard Google Drive filters for 'Tipo', 'Personas', 'Modificado', and 'Fuente'.

**En tu computador  
podrás ver la  
siguiente carpeta.**

**Descarga el archivo  
**CREA-TU-PROPIO-  
VIDEOJUEGO.zip****



# Para iniciar

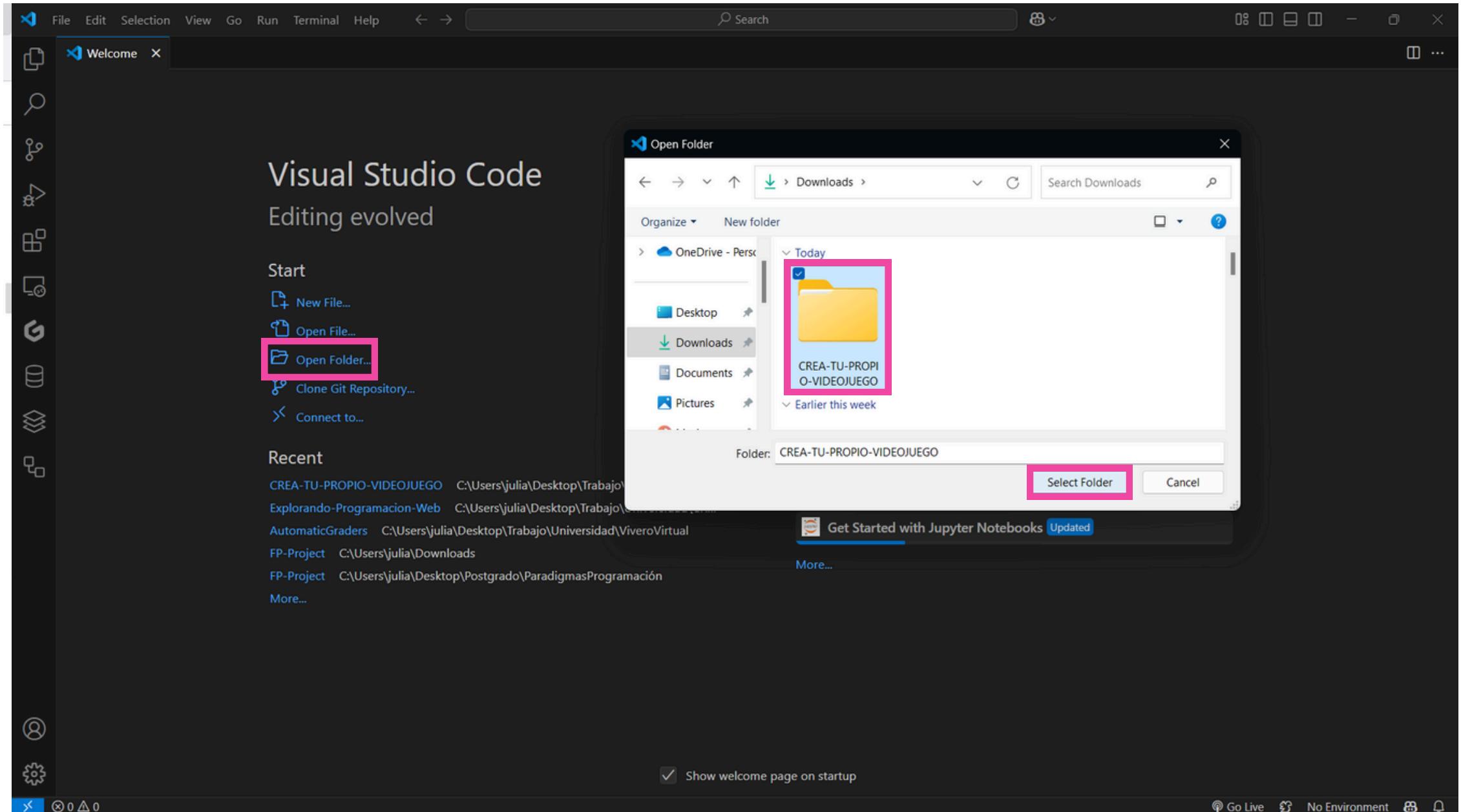


**Descomprime el  
archivo que  
descargaste en el  
paso anterior.**

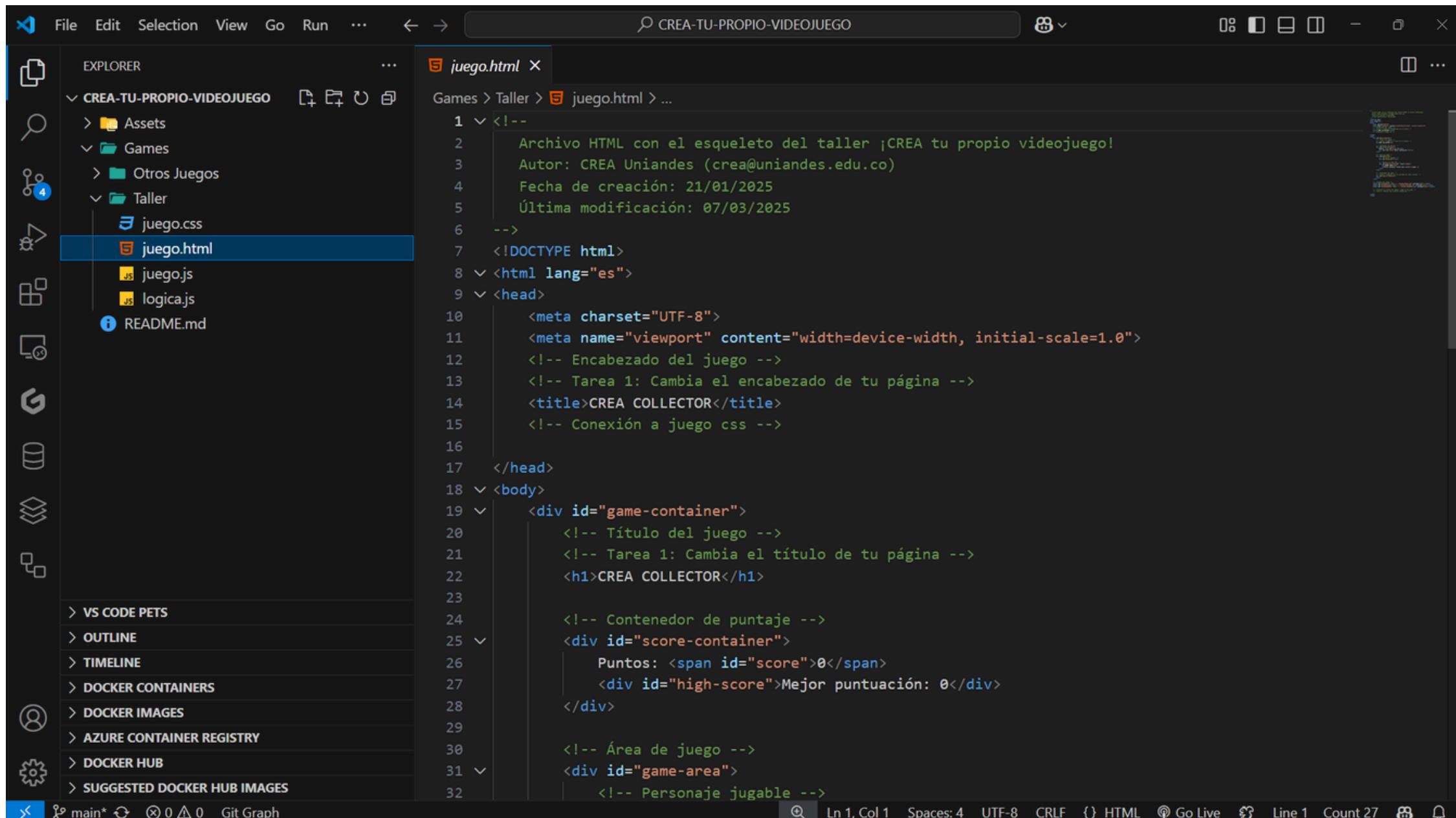
Tendrás la carpeta  
**CREA-TU-PROPIO-  
VIDEOJUEGO**



# Para iniciar



Dentro de **Visual Studio Code**, abre la carpeta **CREA-TU-PROPIO-VIDEOJUEGO**. La misma del paso anterior.



The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a project structure under "CREA-TU-PROPIO-VIDEOJUEGO".
  - Assets
  - Games (selected)
  - Otros Juegos
  - Taller (selected)- Files listed in the Taller folder:
  - juego.css
  - juego.html (highlighted)
  - juegojs
  - logica.js
- README.md

**Code Editor:** The "juego.html" file is open, displaying the following code:

```
1 <!--
2 Archivo HTML con el esqueleto del taller ¡CREA tu propio videojuego!
3 Autor: CREA Uniandes (crea@uniandes.edu.co)
4 Fecha de creación: 21/01/2025
5 Última modificación: 07/03/2025
6 -->
7 <!DOCTYPE html>
8 <html lang="es">
9 <head>
10 <meta charset="UTF-8">
11 <meta name="viewport" content="width=device-width, initial-scale=1.0">
12 <!-- Encabezado del juego -->
13 <!-- Tarea 1: Cambia el encabezado de tu página -->
14 <title>CREA COLLECTOR</title>
15 <!-- Conexión a juego css -->
16
17 </head>
18 <body>
19 <div id="game-container">
20 <!-- Título del juego -->
21 <!-- Tarea 1: Cambia el título de tu página -->
22 <h1>CREA COLLECTOR</h1>
23
24 <!-- Contenedor de puntaje -->
25 <div id="score-container">
26 Puntos: <span id="score">0</span>
27 <div id="high-score">Mejor puntuación: 0</div>
28 </div>
29
30 <!-- Área de juego -->
31 <div id="game-area">
32 <!-- Personaje jugable -->
```

**El archivo logica.js contiene las funciones más complejas de nuestro juego. Por lo tanto, no vamos a modificarlo durante el taller.**

**Vamos a desarrollar todas nuestras actividades dentro de la carpeta **Taller**.**

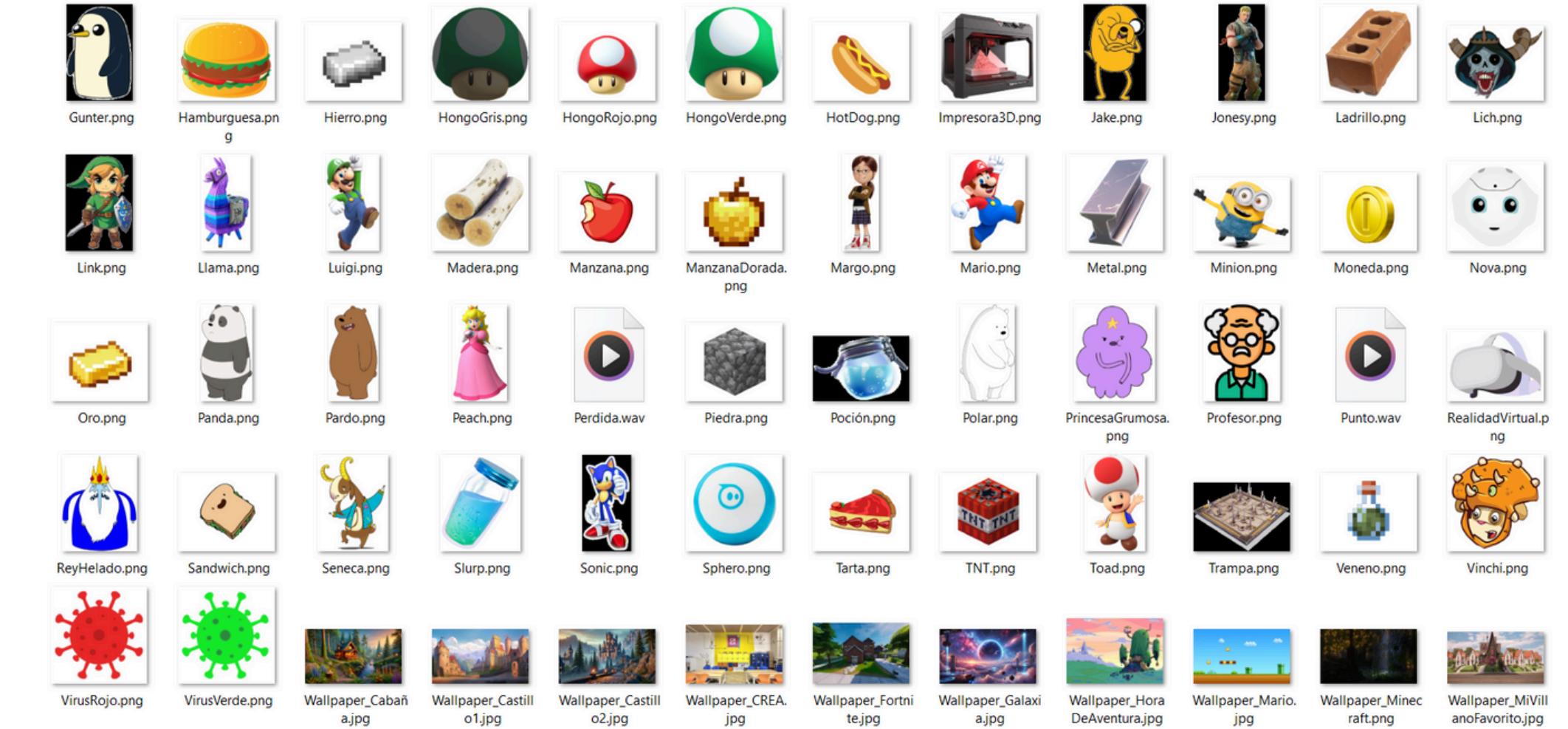
**Dentro de esta, veremos 4 archivos juego.css, juego.html, juego.js y logica.js**

# ¿Qué hay dentro de nuestro proyecto?



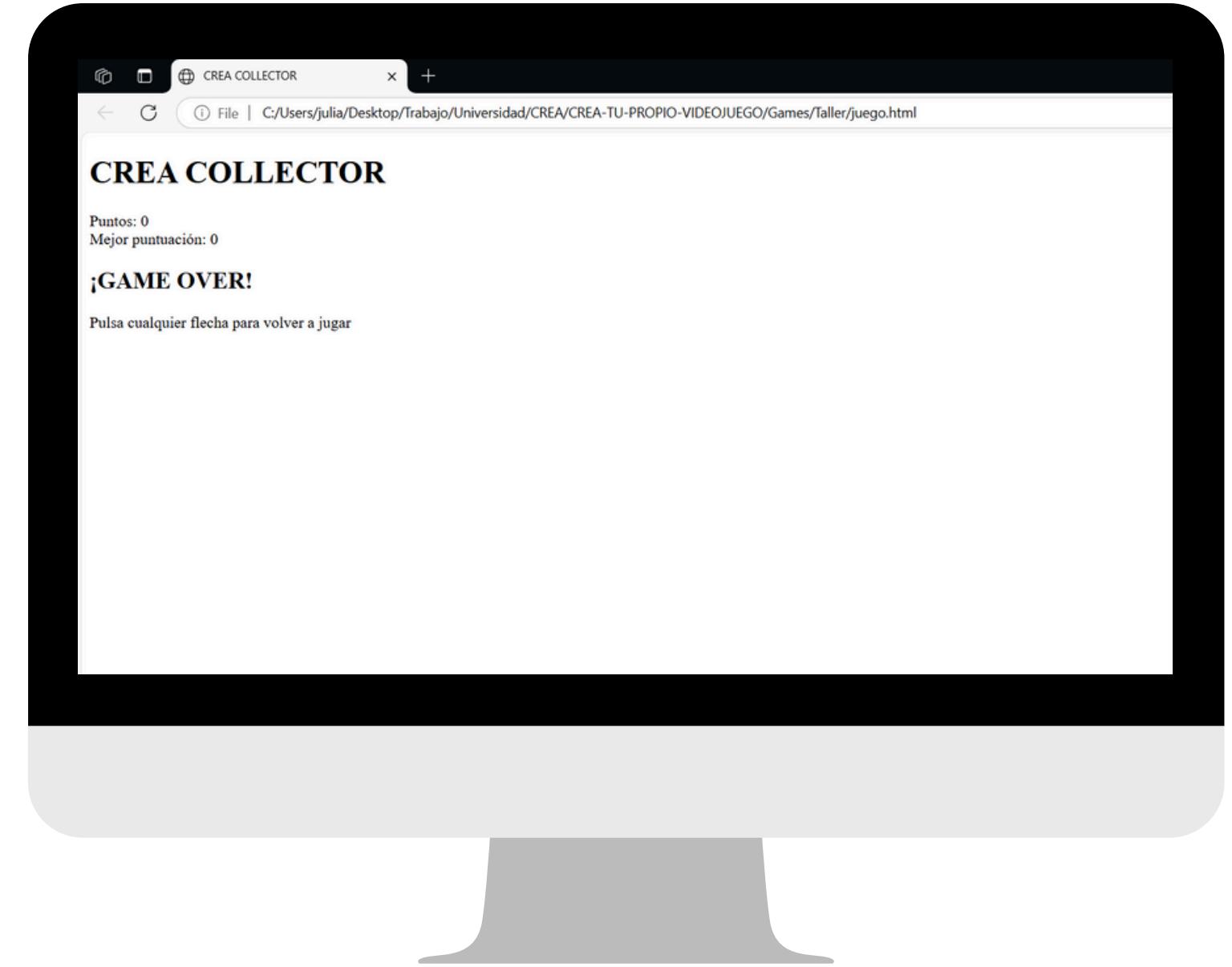
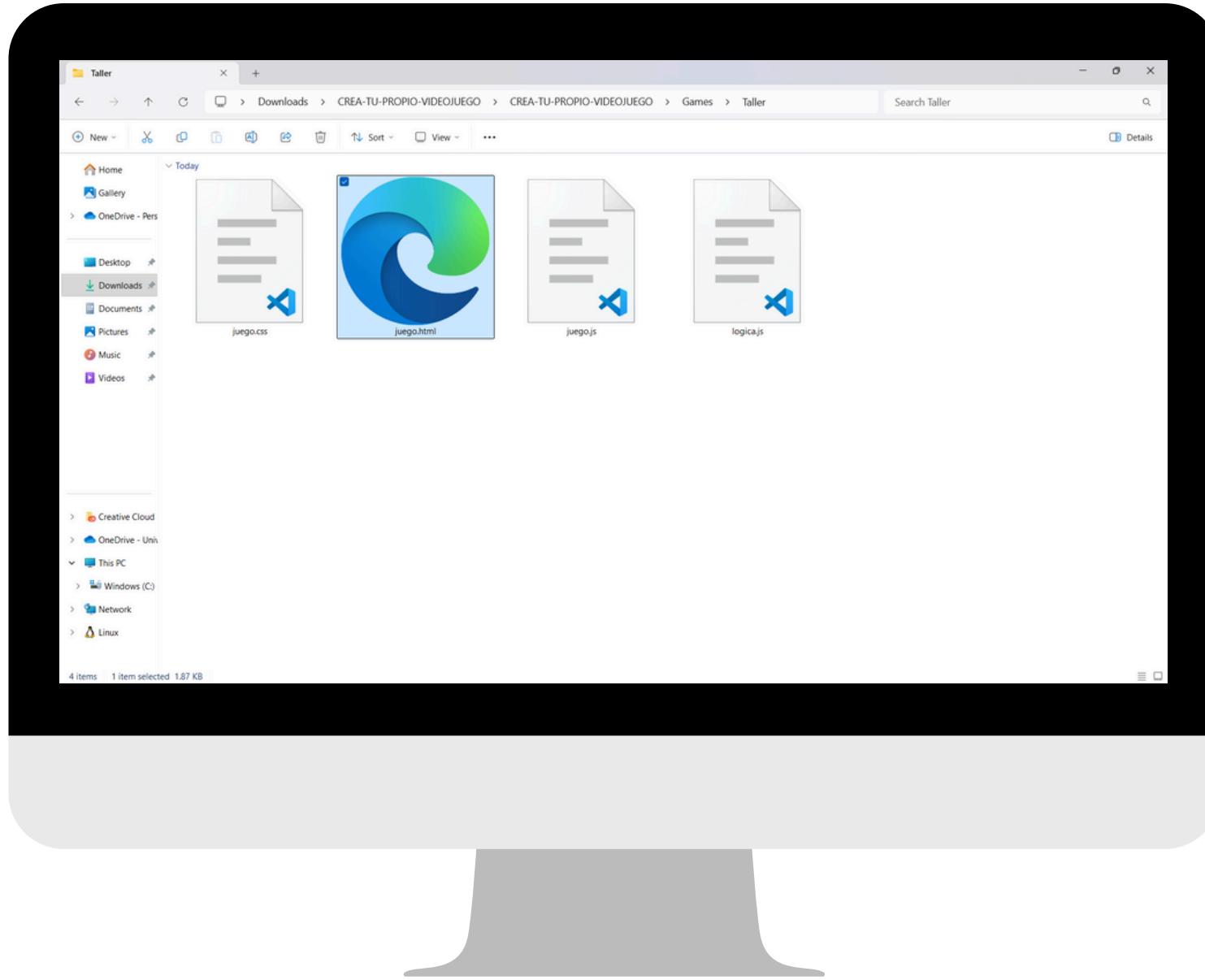
## Carpeta Assets

Cuando quieras usar una de las imágenes debes llamarla de la siguiente forma:



'.../Assets/Gunter.png'

# ¿Cómo probar nuestro juego?



Abre el archivo **juego.html** en el explorador,  
haciendo doble click.



# Conceptos básicos

¡**iCREA** tu Propio Videojuego!



# ¿Qué es HTML?

- Define la estructura de una página web.
- Funciona con contenedores.
- **<head>**: Provee información general, como el título.
- **<body>**: Representa el contenido de la página web.
- **<div>**: Contenedor genérico. Se usa para agrupar contenido.
- **id**: Identifica elementos únicos.

## HyperText Markup Language

```
Games > Taller > juego.html > html > body > div#game-container
1  <!--
2   Archivo HTML con el esqueleto del taller ¡CREA tu propio videojuego!
3   Autor: CREA Uniandes (crea@uniandes.edu.co)
4   Fecha de creación: 21/01/2025
5   Última modificación: 07/03/2025
6  -->
7  <!DOCTYPE html>
8  <html lang="es">
9   <head>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12    <!-- Encabezado del juego -->
13    <!-- Tarea 1: Cambia el encabezado de tu página -->
14    <title>CREA COLLECTOR</title>
15    <!-- Conexión a juego css -->
16
17   </head>
18   <body>
19    <div id="game-container">
20      <!-- Título del juego -->
21      <!-- Tarea 1: Cambia el título de tu página -->
22      <h1>CREA COLLECTOR</h1>
23
24      <!-- Contenedor de puntaje -->
25      <div id="score-container">
26          Puntos: <span id="score">0</span>
27          <div id="high-score">Mejor puntuación: 0</div>
28      </div>
29
30      <!-- Área de juego -->
31      <div id="game-area">
32          <!-- Personaje jugable -->
33          <div id="character"></div>
34
```



# TAREA 1:

**Hagamos algo sencillo...**

**En el archivo *juego.html*, cambia el nombre de la página web y el título de tu juego**

```
juego.html X
Games > Taller > juego.html > html > body > div#game-container > div#score-container

1  <!--
2   Archivo HTML con el esqueleto del taller ¡CREA tu propio videojuego!
3   Autor: CREA Uniandes (crea@uniandes.edu.co)
4   Fecha de creación: 21/01/2025
5   Última modificación: 07/03/2025
6  -->
7  <!DOCTYPE html>
8  <html lang="es">
9  <head>
10 <meta charset="UTF-8">
11 <meta name="viewport" content="width=device-width, initial-scale=1.0">
12 <!-- Encabezado del juego -->
13 <!-- Tarea 1: Cambia el encabezado de tu página -->
14 <title>CREA COLLECTOR</title> ←
15 <!-- Conexión a juego css -->
16
17 </head>
18 <body>
19 <div id="game-container">
20   <!-- Título del juego -->
21   <!-- Tarea 1: Cambia el título de tu página -->
22   <h1>CREA COLLECTOR</h1> ←
```

**Recuerda guardar siempre que realices un cambio para que tu proyecto se actualice (ctrl+s)**

# TAREA 1:

Hagamos algo sencillo...

En el archivo **juego.html**, cambia el nombre de la página web y el título de tu juego

## Solución

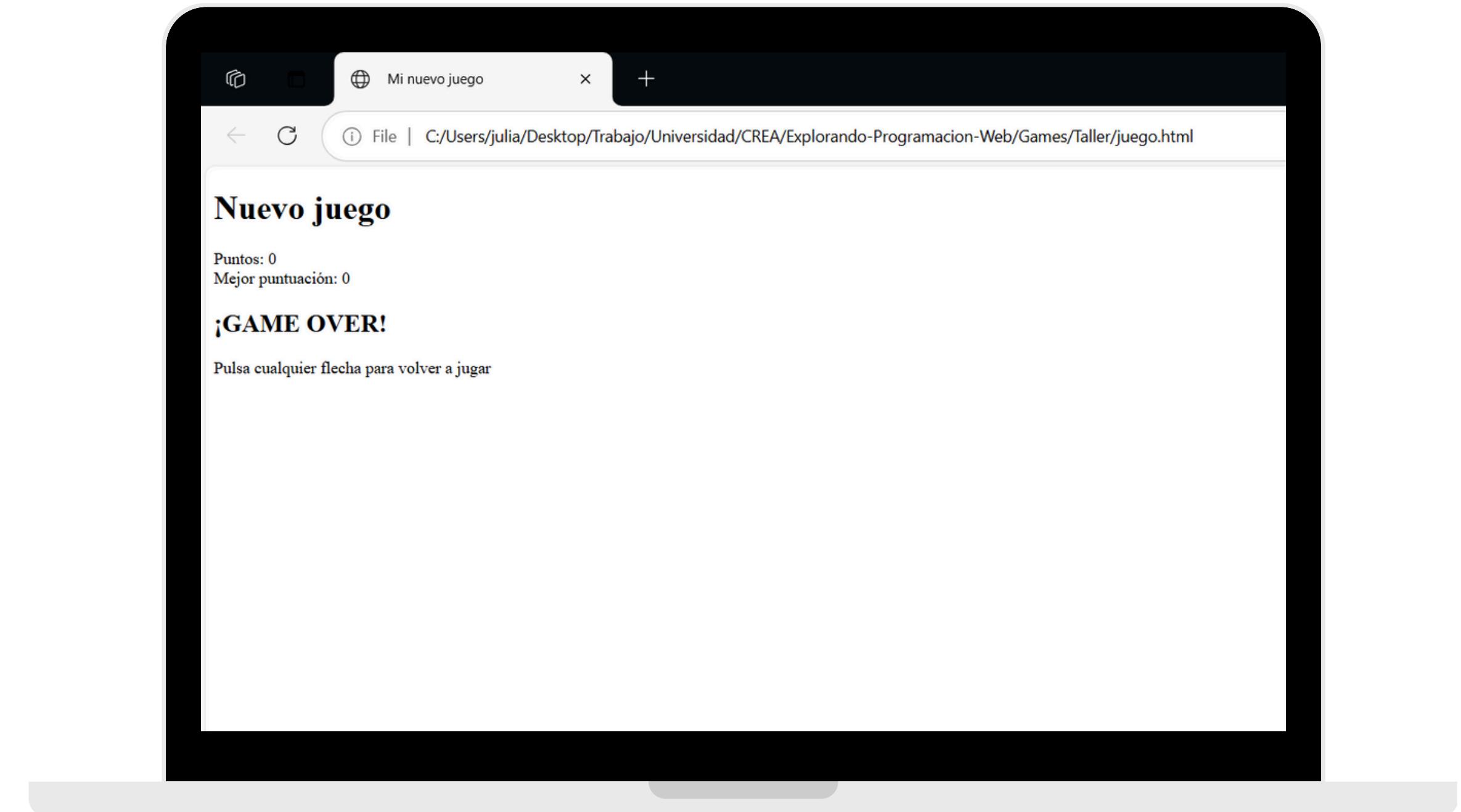
```
10    <meta charset="UTF-8" />
11    <meta name="viewport" content="width=device-width, i
12    <!-- Encabezado del juego -->
13    <!-- Tarea 1: Cambia el encabezado de tu página -->
14    <title>Nuevo Juego</title> ←
15    <!-- Conexión a juego css -->
16  </head>
17  <body>
18    <div id="game-container">
19      <!-- Título del juego -->
20      <!-- Tarea 1: Cambia el título de tu página -->
21      <h1>Mi nuevo Juego</h1> ←
22      <!-- Contenedor de puntaje -->
23      <div id="score-container">
24        Puntos: <span id="score">0</span>
```

# TAREA 1:

**Hagamos algo sencillo...**

**En el archivo *juego.html*, cambia el nombre de la página web y el título de tu juego**

## Solución



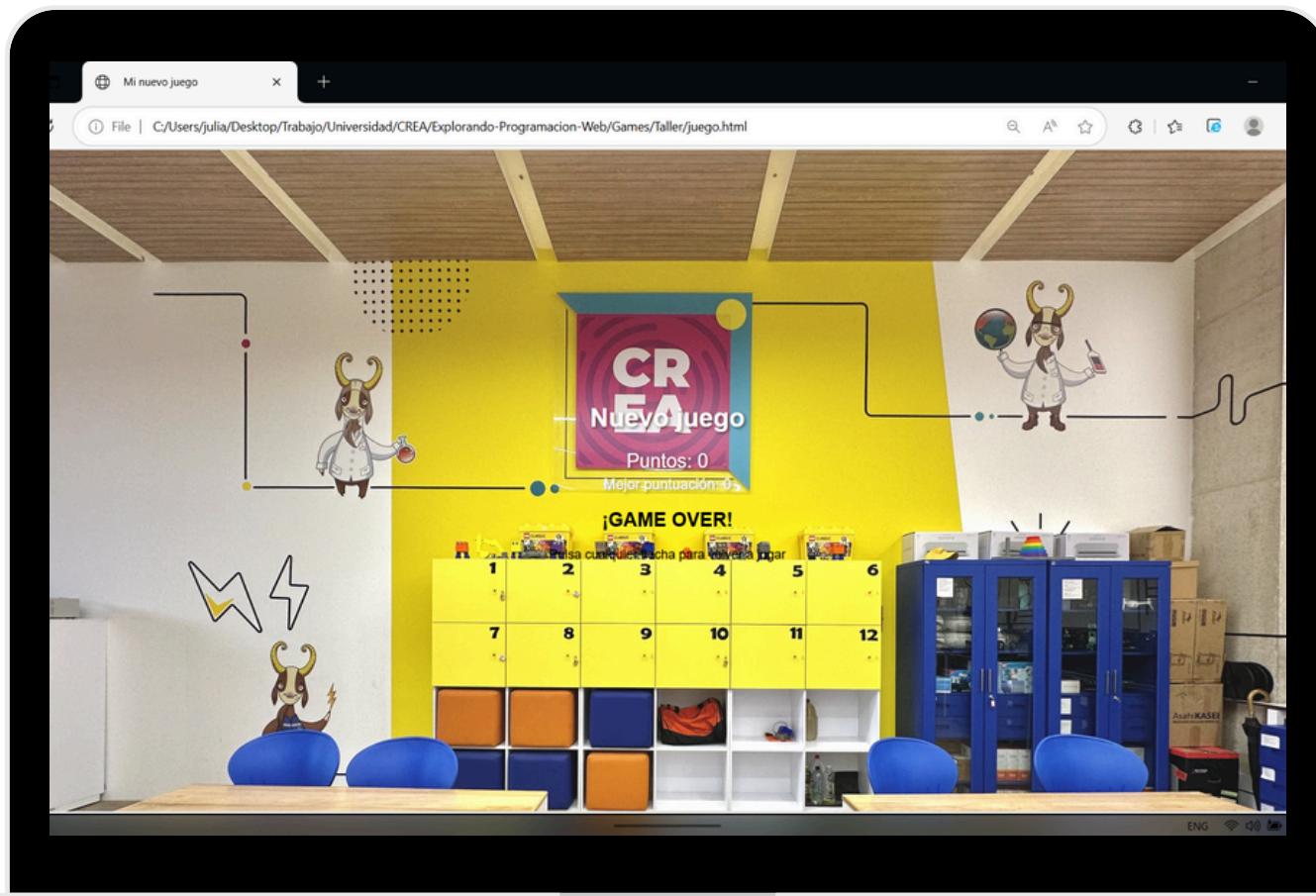
# ¿Qué es CSS?

```

Games > Taller > juego.css > ...
4  * Fecha de creación: 21/01/2025
5  * Última modificación: 07/03/2025
6  */
7
8  /* Estilos generales para la página */
9  body {
10    margin: 0;
11    padding: 0;
12    height: 100vh;
13    display: flex;
14    justify-content: center;
15    align-items: center;
16    font-family: Arial, sans-serif;
17    background-size: cover;
18    background-image: url(' ../../Assets/Wallpaper_CREA.jpg'); /* Tarea 2: Cambia la imagen de fondo */
19  }
20
21  /* Contenedor principal del juego */
22  #game-container {
23    text-align: center;
24  }
25
26  /* Estilo para el título del juego */
27  h1 {
28    color: white;
29    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
30  }
31
32  /* Estilo para el contador de puntos */
33  #score-container {
34    font-size: 24px;
35    color: white;
36    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
37    margin: 20px 0;
38  }
39
40  /* Estilo para el contador de mejor puntaje */
41  #high-score {
42    font-size: 18px;
43    margin-top: 5px;
44  }
45
46  /* Área de juego donde ocurre la acción */
47  #game-area {
48    width: 800px;
49    height: 600px;
50    position: relative;

```

# Cascading Style Sheets

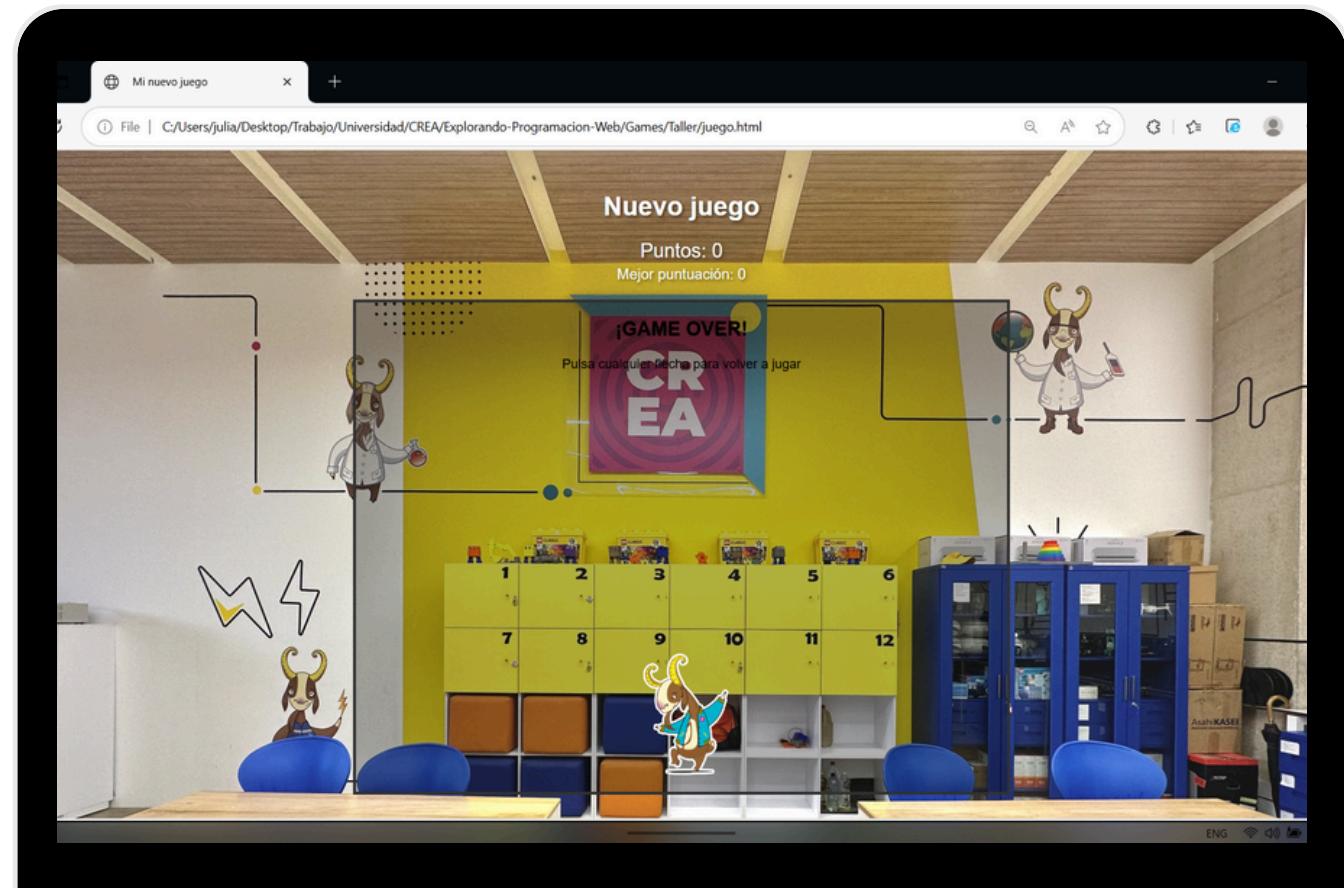


**En este archivo se definen los estilos del título, el fondo y los demás elementos visuales que contenga la página web**

# ¿Qué es CSS?

```
43 /* Área de juego donde ocurre la acción */
44 #game-area {
45     width: 800px;
46     height: 600px;
47     position: relative;
48     border: 4px solid #333;
49     overflow: hidden;
50     background-color: #rgba(0, 0, 0, 0.3);
51 }
52
53 /* Estilo para el personaje controlado por el jugador */
54 #character {
55     width: 150px;
56     height: 150px;
57     position: absolute;
58     bottom: 20px;
59     left: 350px;
60     background-image: url('../Assets/Taller/Uniandes_Seneca.png'); /* Tarea 2: Cambia la imagen de fondo */
61     background-size: contain;
62     background-repeat: no-repeat;
63 }
64
```

# Área de juego y Personaje





# TAREA 2:

Hagamos algo sencillo...

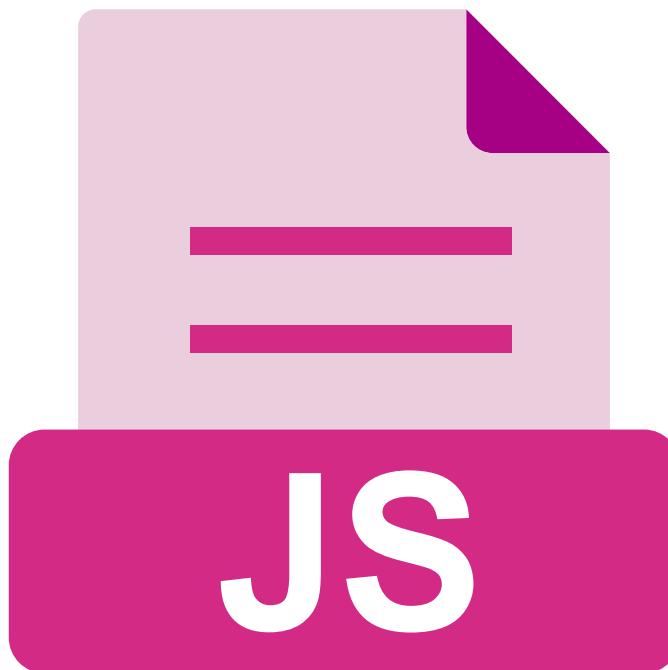
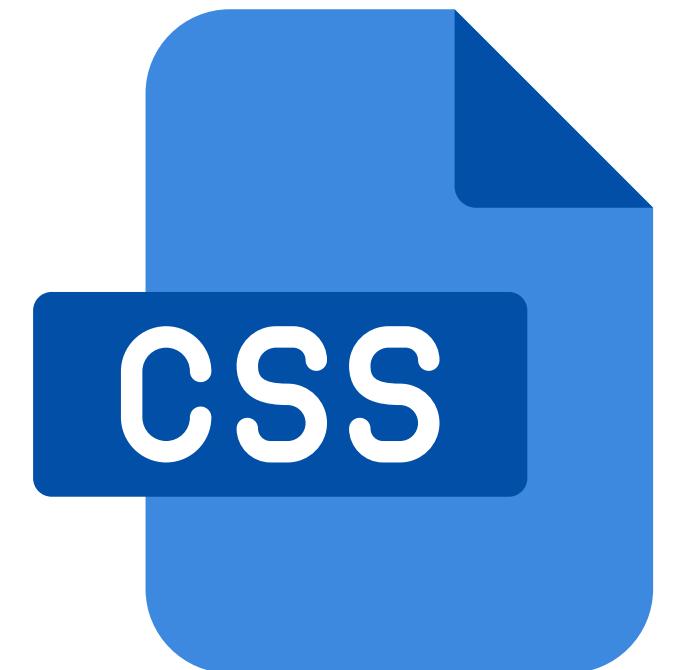
En el archivo **juego.css**, cambia el la imagen del fondo y la imagen del personaje

```
# juego.css •  
Games > Taller > # juego.css > ...  
9  body {  
10 margin: 0;  
11 padding: 0;  
12 height: 100vh;  
13 display: flex;  
14 justify-content: center;  
15 align-items: center;  
16 font-family: Arial, sans-serif;  
17 background-size: cover;  
18 /* Tarea 2: Cambia la imagen de fondo */  
19 background-image: url('../Assets/Wallpaper_CREA.jpg'); ←  
58 character {  
59 width: 150px;  
60 height: 150px;  
61 position: absolute;  
62 bottom: 20px;  
63 left: 350px;  
64 /* Tarea 2: Cambia la imagen del personaje */  
65 background-image: url('../Assets/Seneca.png'); ←  
66 background-size: contain;  
67 background-repeat: no-repeat;  
68 }
```

Recuerda guardar siempre que realices un cambio para que tu proyecto se actualice (ctrl+s)

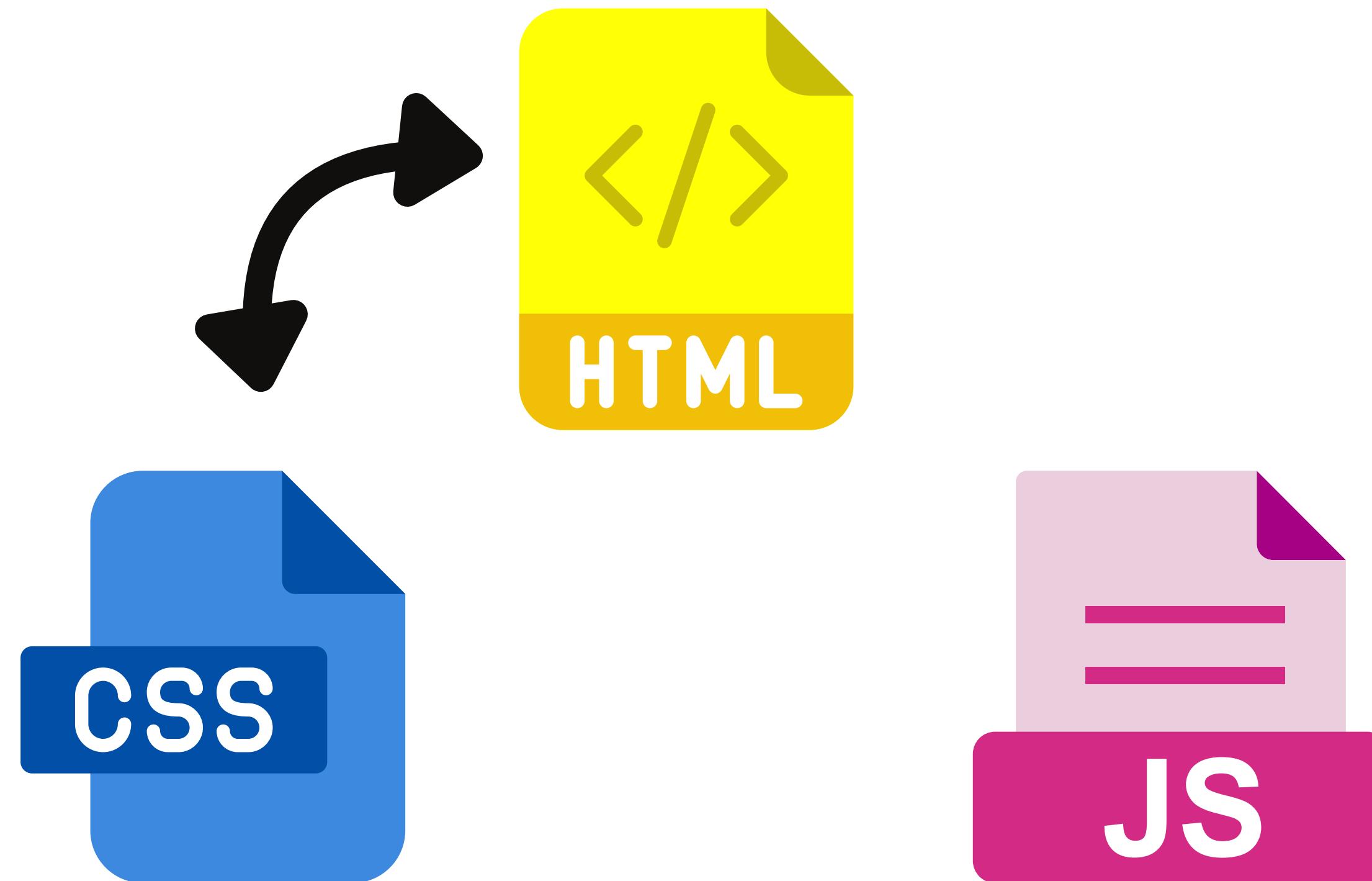
# ¿Por qué no se ve?

Ningún archivo está conectado a los demás

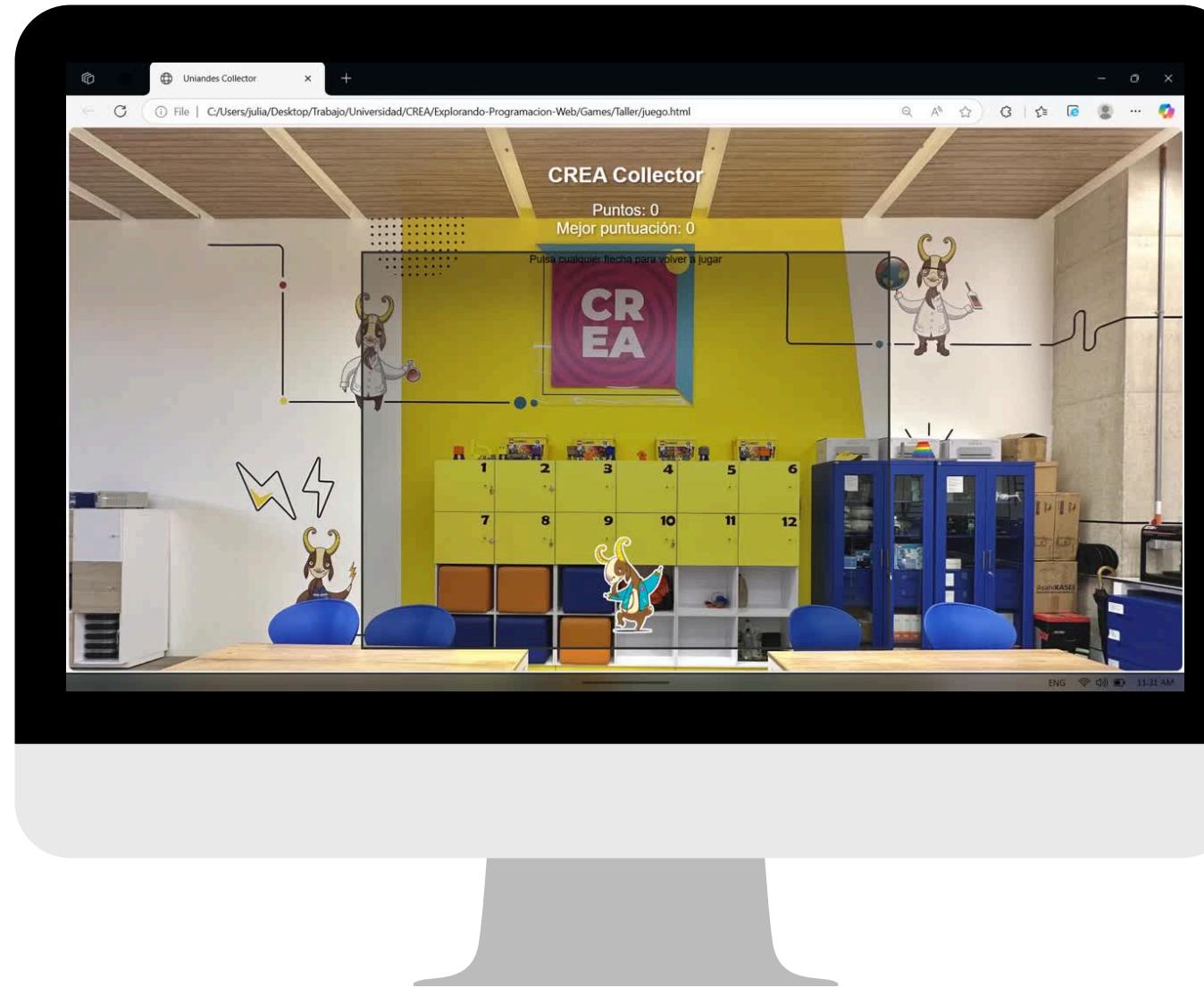


# ¿Por qué no se ve?

Vamos a conectar el HTML y el CSS



# ¿Cómo conectar mi CSS?



```
11      <meta name="viewport" content="width=device-
12      <!-- Encabezado del juego --&gt;
13      <!-- Tarea 1: Cambia el encabezado de tu pág
14      &lt;title&gt;Nuevo Juego&lt;/title&gt;
15      <!-- Conexión a juego CSS --&gt;
16      &lt;link rel="stylesheet" href="juego.css"&gt; ←
17      &lt;/head&gt;
18      &lt;body&gt;
19          &lt;div id="game-container"&gt;
20              <!-- Título del juego --&gt;
21              <!-- Tarea 1: Cambia el título de tu pág
22              &lt;h1&gt;Mi nuevo Juego&lt;/h1&gt;
23
24              <!-- Contenedor de puntaje --&gt;
25              &lt;div id="score-container"&gt;
26                  Puntos: &lt;span id="score"&gt;0&lt;/span&gt;
27                  &lt;div id="high-score"&gt;Mejor puntuació</pre>
```

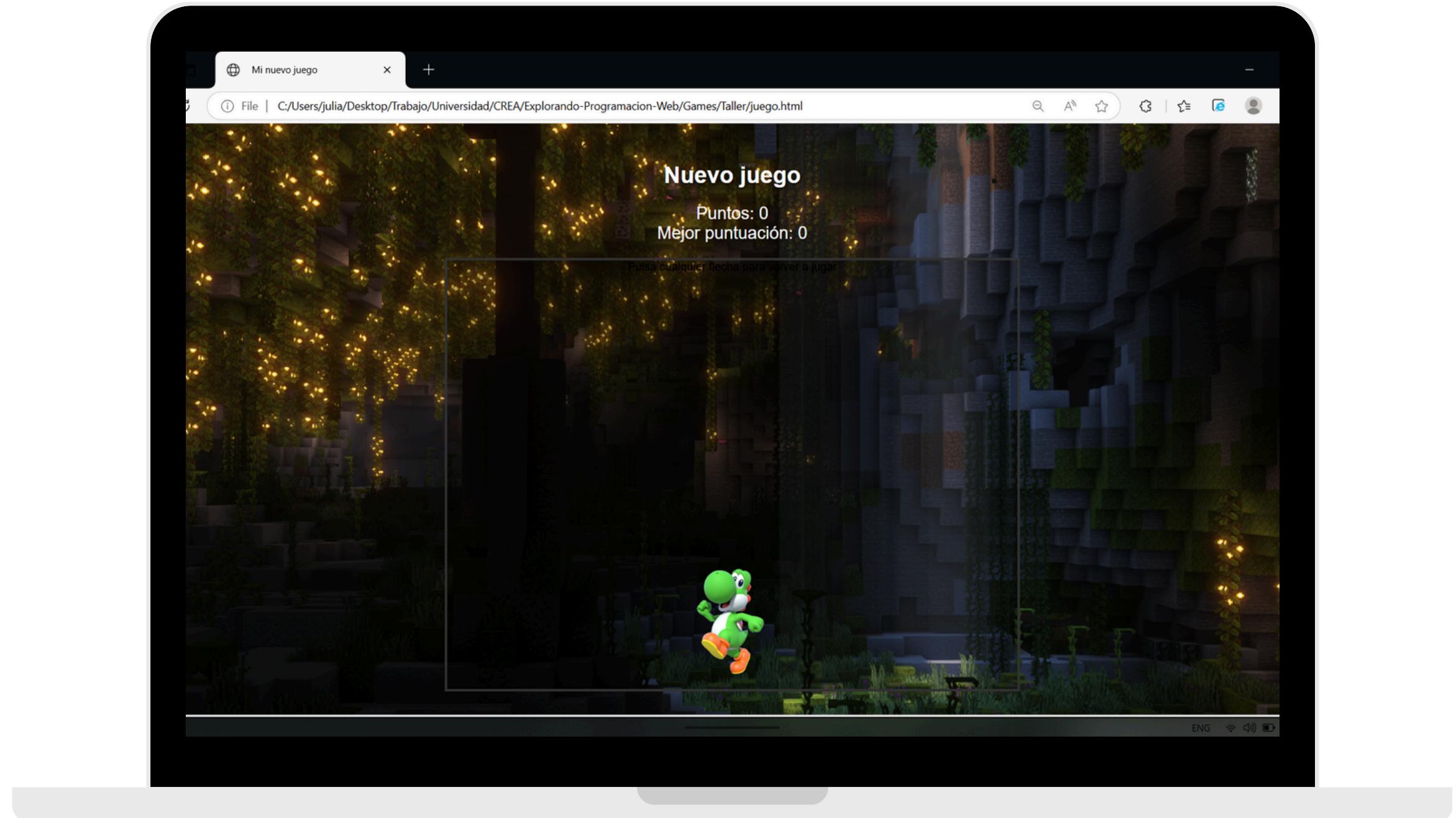
<link rel="stylesheet" href="juego.css" >

# ¿Qué es CSS?



Hagamos algo sencillo...

Cambia la imagen de fondo y el personaje principal



# ¿Qué es JavaScript? ¡Démosle Vida a la Web!



- **JavaScript permite la interactividad.**
- **Controla el comportamiento de la página web.**
- **Ejecuta acciones en el navegador del usuario.**



# ¿Qué es JavaScript?

## Conceptos

- **VARIABLES:** Almacenan datos que pueden cambiar a lo largo del programa.
- **CONSTANTES:** Almacenan datos que permanecen fijos durante toda la ejecución del programa.

```
9 // Variables globales del juego
10 let objetosEnJuego = []; // Lista de objetos activos en el juego
11 let gameArea = null;
12 let personaje = null;
13 let scoreDisplay = null;
14 let highScoreDisplay = null;
15 let heartsContainer = null;
16 let gameOverMessage = null;
17
18 // Variables de estado del juego
19 let score = 0; // Puntuación actual
20 let highScore = 0; // Mejor puntuación
21 let gameOver = false; // Estado de fin de juego
22 let vidas = 3; // Número de vidas
23 let intervaloCaida; // Intervalo para mover objetos
24 let intervaloCreacion; // Intervalo para crear objetos
25
26 // Sonidos
27 let pointSound;
28 let looseSound;
29 let gameOverSound;
30
31 // Constantes de juego
32 const ANCHO_JUEGO = 800;
33 const ALTO_JUEGO = 600;
34 const ANCHO_PERSONAJE = 110;
35 const ALTO_PERSONAJE = 150;
36
```

Vamos a analizar el archivo logica.js. Sin embargo este no lo vamos a modificar porque contiene funciones avanzadas para la interacción del juego.

# ¿Qué es JavaScript?

## Conceptos

- **Funciones:** Las funciones agrupan código reutilizable para ejecutar acciones específicas.

¿Qué hace esta función?

```
481 function reiniciarJuego() {  
482     console.log("Reiniciando juego");  
483  
484     // Limpiar objetos existentes  
485     while (objetosEnJuego.length > 0) {  
486         if (objetosEnJuego[0].parentNode) {  
487             gameArea.removeChild(objetosEnJuego[0]);  
488         }  
489         objetosEnJuego.shift();  
490     }  
491  
492     // Reiniciar variables  
493     score = 0;  
494     gameOver = false;  
495     if (scoreDisplay) {  
496         scoreDisplay.textContent = score;  
497     }  
498     if (gameOverMessage) {  
499         gameOverMessage.classList.add('hidden');  
500     }  
501  
502     // Reiniciar vidas  
503     vidas = VIDAS_INICIALES;  
504  
505     // Actualizar los corazones  
506     actualizarVidasVisualmente();  
507  
508     // Reiniciar posición del personaje  
509     if (personaje) {  
510         personaje.style.left = "350px";  
511     }  
512  
513     // Iniciar el juego de nuevo  
514     iniciarIntervalos();  
515 }
```



# ¿Qué es JavaScript?

## Conceptos

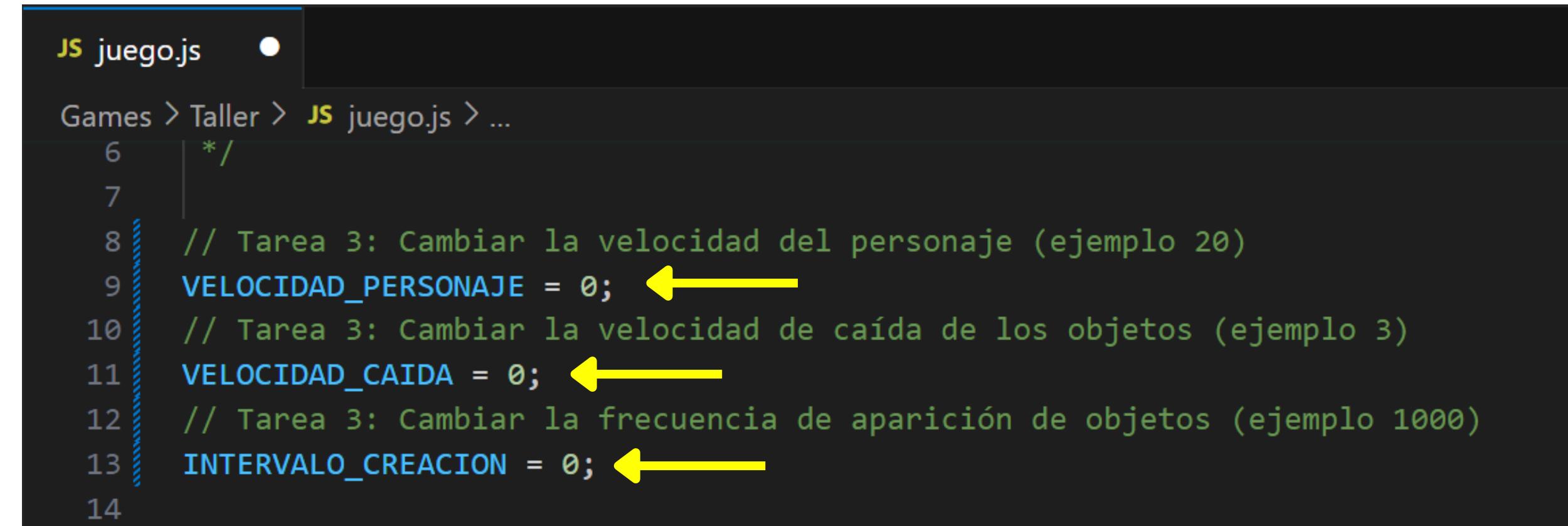


- **Eventos:** Los eventos permiten que una página web reaccione a acciones del usuario, como presionar teclas, hacer clic o mover el mouse.

```
144 function configurarControles(velocidad) {  
145   document.addEventListener('keydown', function(event) { ←  
146     // Verificar que el personaje existe  
147     if (!personaje) {  
148       console.error("Error: Personaje no encontrado");  
149       return;  
150     }  
151  
152     // Si el juego terminó, reiniciar con cualquier flecha  
153     if (gameOver && (event.key === 'ArrowLeft' || event.key === 'ArrowRight')) { ←  
154       reiniciarJuego(); ←  
155       return;  
156     }  
157  
158     // Obtener posición actual del personaje de manera más robusta  
159     let posX;  
160     // Si el personaje ya tiene estilo left definido, usarlo  
161     if (personaje.style.left) {  
162       posX = parseInt(personaje.style.left);  
163     }  
164     // Si no, obtener el valor computado  
165     else {  
166       const computedStyle = window.getComputedStyle(personaje);  
167       posX = parseInt(computedStyle.left);  
168     }  
169  
170     // Si no se pudo obtener un valor válido, usar el valor predeterminado  
171     if (isNaN(posX)) {  
172       posX = 350;  
173     }  
174  
175     // Ancho del área de juego y del personaje  
176     const anchoJuego = gameArea.offsetWidth;  
177     const anchoPersonaje = personaje.offsetWidth;  
178  
179     // Mover a la izquierda o derecha según la tecla presionada ←  
180     if (event.key === 'ArrowLeft') { ←  
181       posX = Math.max(0, posX - velocidad);  
182     } else if (event.key === 'ArrowRight') { ←  
183       posX = Math.min(anchoJuego - anchoPersonaje, posX + velocidad);  
184     }  
185  
186     // Establecer la nueva posición  
187     personaje.style.left = posX + 'px';  
188   });  
189 }
```

# TAREA 3:

Hagamos algo sencillo para que nuestro juego se mueva...



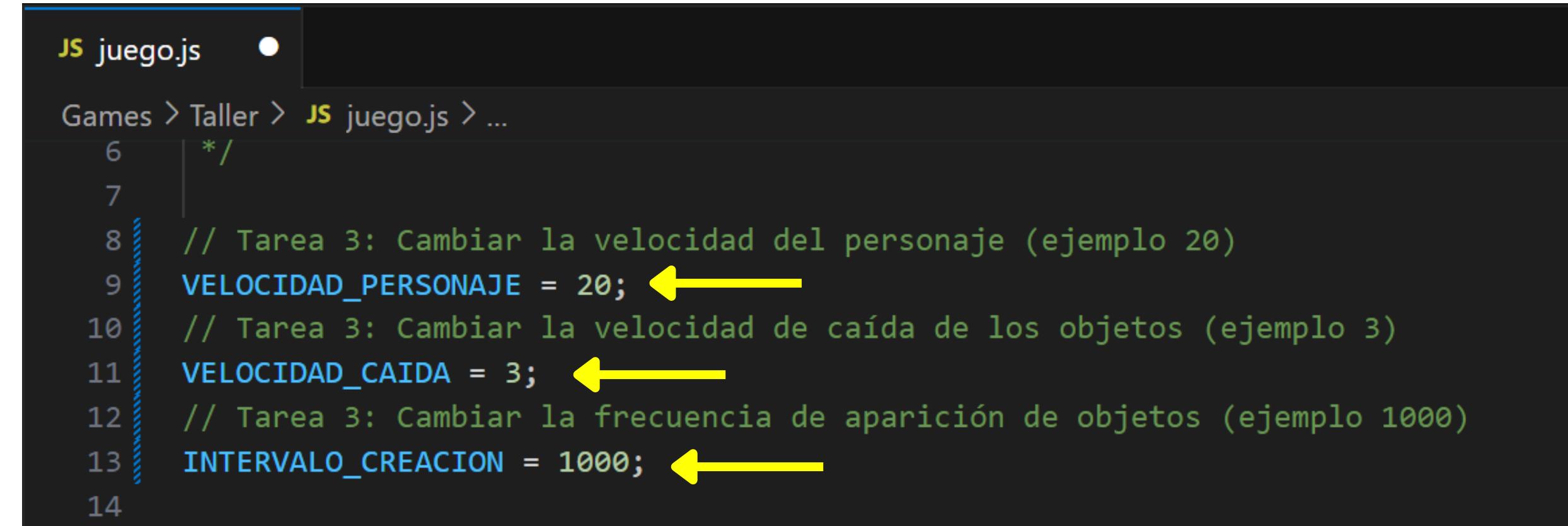
```
JS juego.js
Games > Taller > JS juego.js > ...
6   */
7
8 // Tarea 3: Cambiar la velocidad del personaje (ejemplo 20)
9 VELOCIDAD_PERSONAJE = 0; ←
10 // Tarea 3: Cambiar la velocidad de caída de los objetos (ejemplo 3)
11 VELOCIDAD_CAIDA = 0; ←
12 // Tarea 3: Cambiar la frecuencia de aparición de objetos (ejemplo 1000)
13 INTERVALO_CREACION = 0; ←
14
```

En **juego.js** cambia las variables de velocidad del personaje, caída de objetos y creación de objetos

# TAREA 3:

Hagamos algo sencillo para que nuestro juego se mueva...

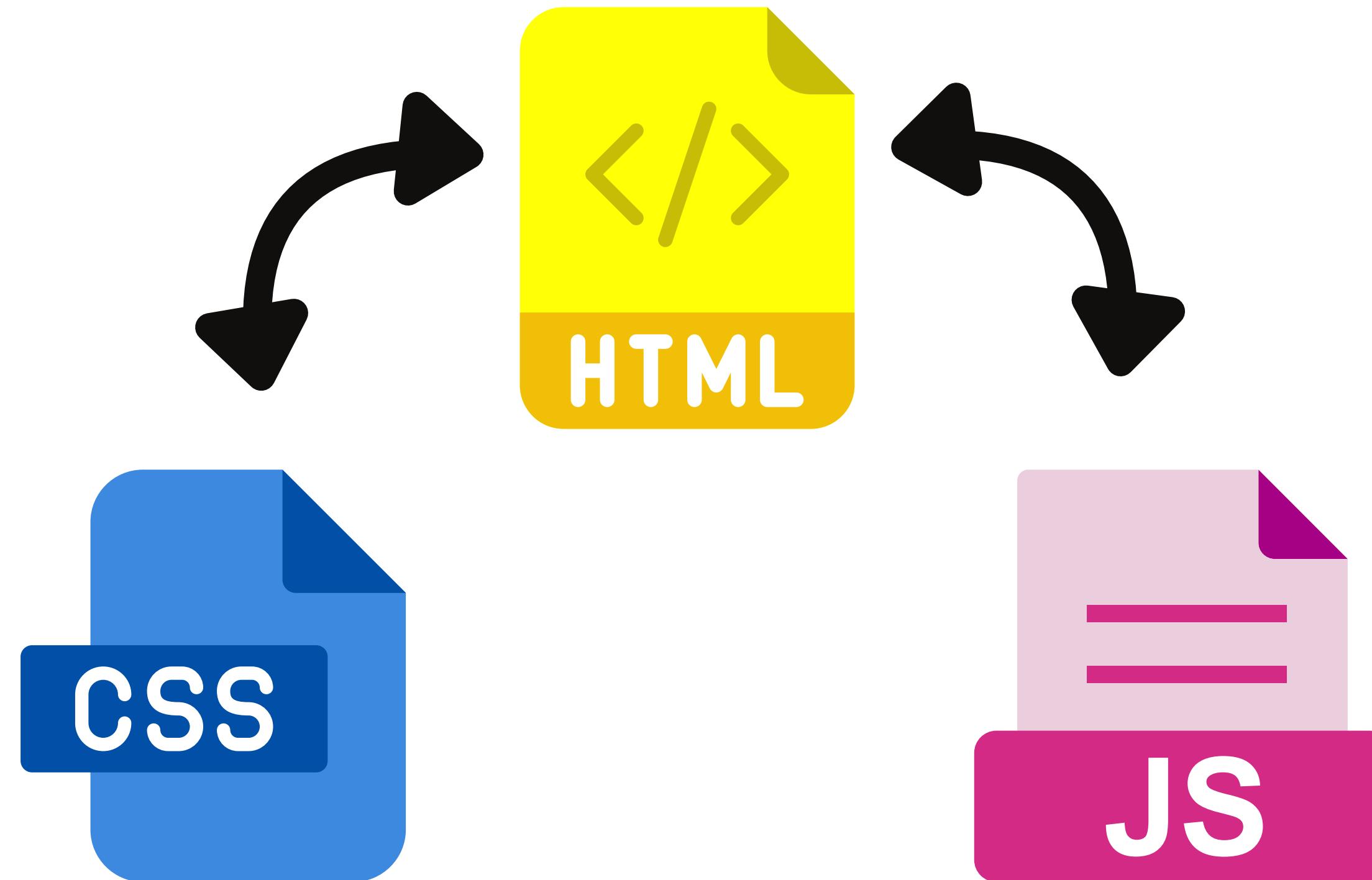
## Solución



```
JS juego.js
Games > Taller > JS juego.js > ...
6   */
7
8 // Tarea 3: Cambiar la velocidad del personaje (ejemplo 20)
9 VELOCIDAD_PERSONAJE = 20; ←
10 // Tarea 3: Cambiar la velocidad de caída de los objetos (ejemplo 3)
11 VELOCIDAD_CAIDA = 3; ←
12 // Tarea 3: Cambiar la frecuencia de aparición de objetos (ejemplo 1000)
13 INTERVALO_CREACION = 1000; ←
14
```

En **juego.js** cambia las variables de velocidad del personaje, caída de objetos y creación de objetos

# Recuerda las conexiones



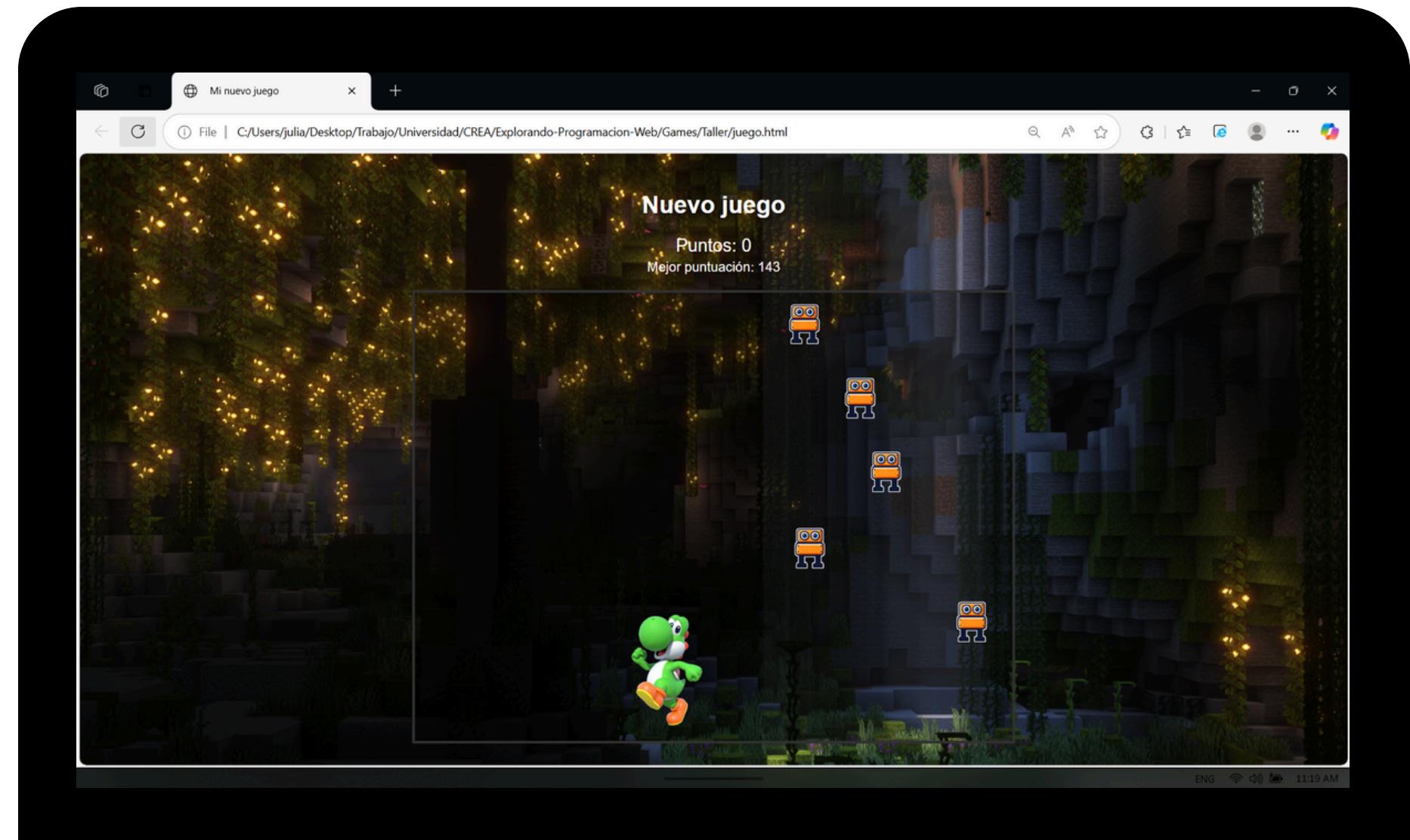
# Vamos a conectar el HTML y el JS

```
40      </div>
41
42
43
44
45
46
47
48      <!-- Sonidos del juego -->
49      <audio id="sonidoPunto" src="../../Assets/Punto.wav" preload="auto">
50      <audio id="sonidoPerdida" src="../../Assets/Perdida.wav" preload="auto">
51      <audio id="sonidoGameOver" src="../../Assets/GameOver.wav" preload="auto">
52
53      <!-- Conexión al archivo de lógica, luego el de juego -->
54      <!-- Tarea 3: Conecta tus archivos JavaScript -->
55      <script src="logica.js"></script> ←
56      <script src="juego.js"></script> ←
57  </body>
58  </html>
59
```

**<script src="logica.js"></script>**  
**<script src="juego.js"></script>**

**Se conectan en ese orden**  
**porque *juego.js* depende**  
**de *logica.js***

# Resultado



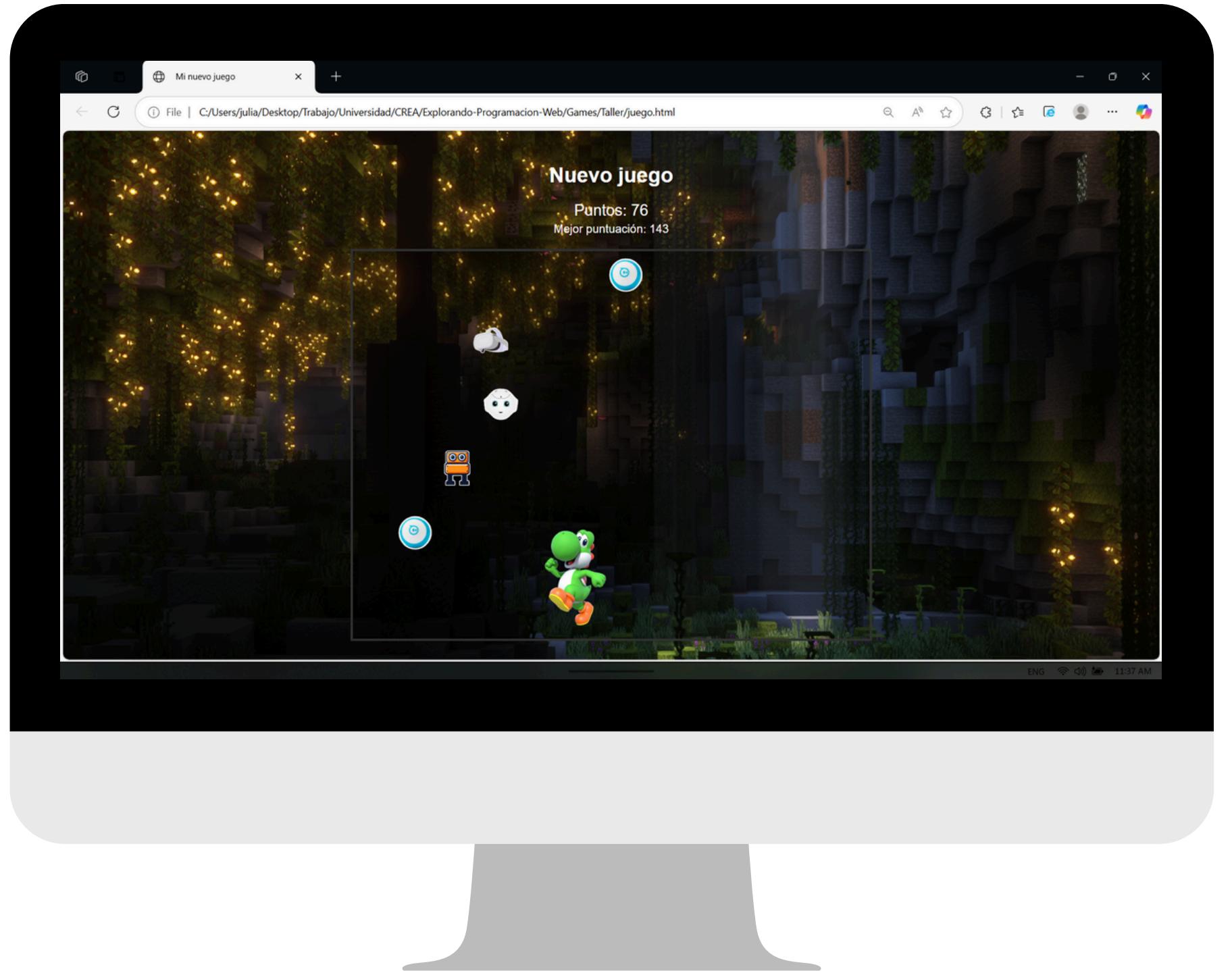


# ¡A programar!

¡**iCREA** tu Propio Videojuego!



# Objetos en caída





**Todos los objetos se  
guardan en una lista**



**Una lista es una estructura de datos  
que nos permite almacenar  
información. Por ejemplo, personas.**

# Así se vería en JavaScript...

[Julián, Ari, Jacobo, Shurys,  
Hugo, Sebas, Laura, Octo]

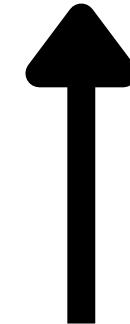




**[Julián, Ari, Jacobo, Shurys,  
Hugo, Sebas, Laura, Octo]**

**Pero solo estamos  
guardando los nombres  
¿Cómo agregar más  
información?**

```
{nombre: Julián,  
edad: 22,  
peliFavorita: Harry Potter,  
animalFavorito: Pingüino}
```



```
[Julián, Ari, Jacobo, Shurys,  
Hugo, Sebas, Laura, Octo]
```

Podemos agrupar toda la información de una persona usando {}

```
[  
{  
  nombre: Julián,  
  edad: 22,  
  peliFavorita: Harry Potter,  
  animalFavorito: Pingüino  
},  
 {  
  nombre: Ari,  
  ...  
},  
 ...  
]
```

The code snippet illustrates a list of objects, likely representing people, stored in an array. Each object has properties: nombre, edad, peliFavorita, and animalFavorito. Two specific objects are highlighted with blue arrows pointing to their names on the right:

- An arrow points from the 'nombre' value 'Julián,' to the name 'Julián,' on the right.
- An arrow points from the 'nombre' value 'Ari,' to the name 'Ari,' on the right.

The names on the right side of the arrows are:  
[Julián,  
 Ari,  
 Jacobo,  
 Shurys,

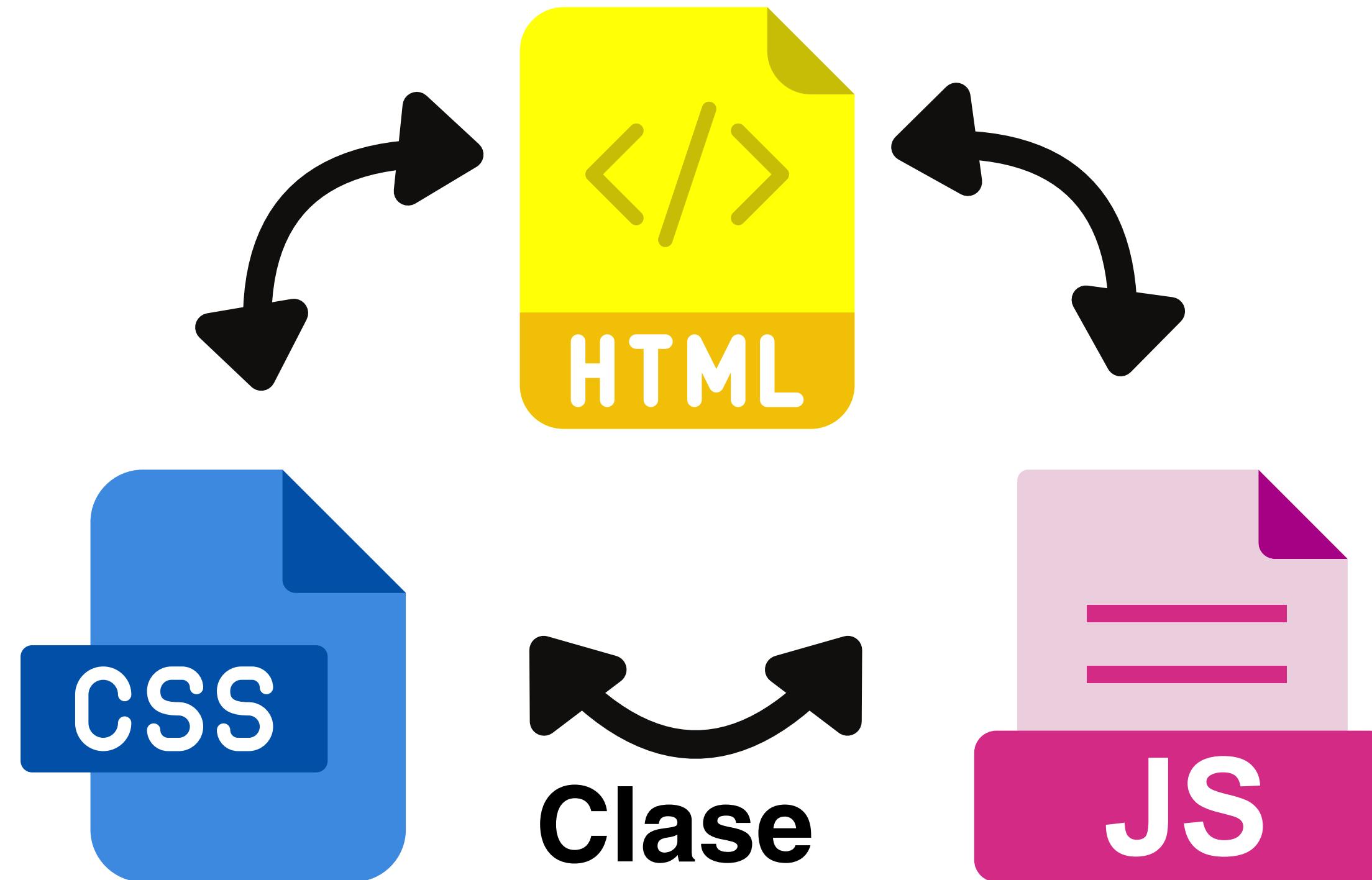
# Nuestros objetos

JS juego.js ● ◊ juego.html

Games > Taller > JS juego.js > ...

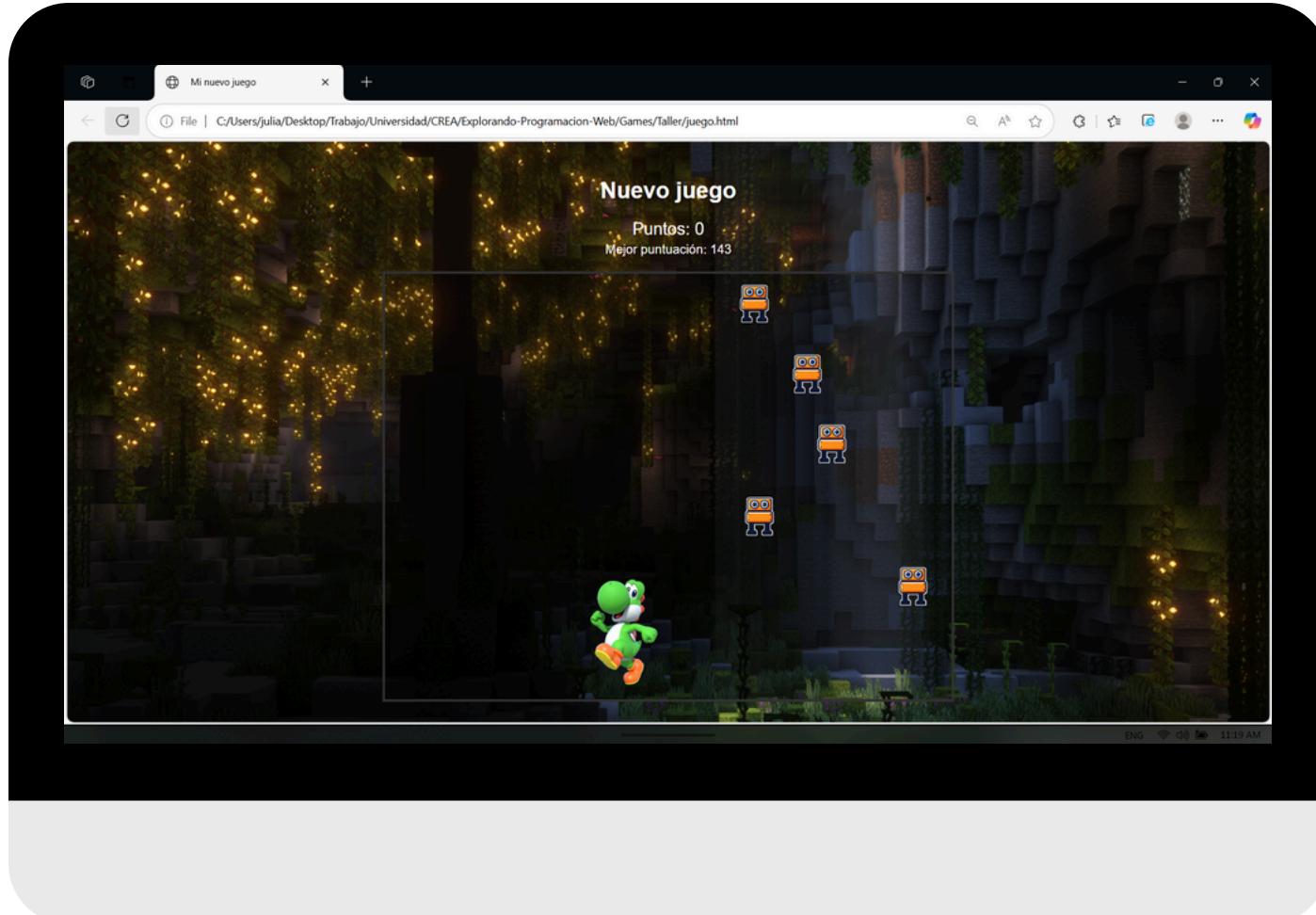
```
19 // La suma de todas las probabilidades debe ser menor o igual a 1
20 OBJETOS = [
21 {
22   tipo: 'bot',           // Nombre del objeto
23   clase: 'objeto-bot',  // Conexión con CSS ←
24   puntos: 1,             // Puntos que da al recogerlo
25   probabilidad: 0.25    // Probabilidad de que aparezca (entre 0 y 1)
26 }
```

# Recuerda las conexiones



# Tarea 4: Crea 4 objetos

## juego.js



```
17 // Cada objeto debe tener: tipo, clase, puntos, y probabilidad
18 // La suma de todas las probabilidades debe ser menor o igual a 1
19 OBJETOS = [
20   {
21     tipo: 'bot',          // Nombre del objeto
22     clase: 'objeto-bot', // Clase CSS para dar estilo
23     puntos: 1,           // Puntos que da al recogerlo
24     probabilidad: 0.25  // Probabilidad de que aparezca (entre 0 y 1)
25   }
26   // Tarea 4: Agrega 3 objetos que den puntos positivos
27 
```

## juego.css

```
78 /* Estilos para los objetos que caen */
79 /* Tarea 4: Agrega 3 objetos que den puntos positivos */
80 .objeto-bot {
81   background-image: url('../Assets/CREABOT.png');
82 }
83 
```

# Objetos en caída

juego.css

```

78  /* Estilos para los objetos que caen */
79  /* Tarea 4: Agrega 3 objetos que den puntos positivos */
80  .objeto-bot {
81    background-image: url('../Assets/CREABOT.png');
82  }
83  .objeto-sphero {
84    background-image: url('../Assets/Sphero.png');
85  }
86  .objeto-vr {
87    background-image: url('../Assets/RealidadVirtual.png');
88  }
89  .objeto-nova {
90    background-image: url('../Assets/Nova.png');
91  }

```

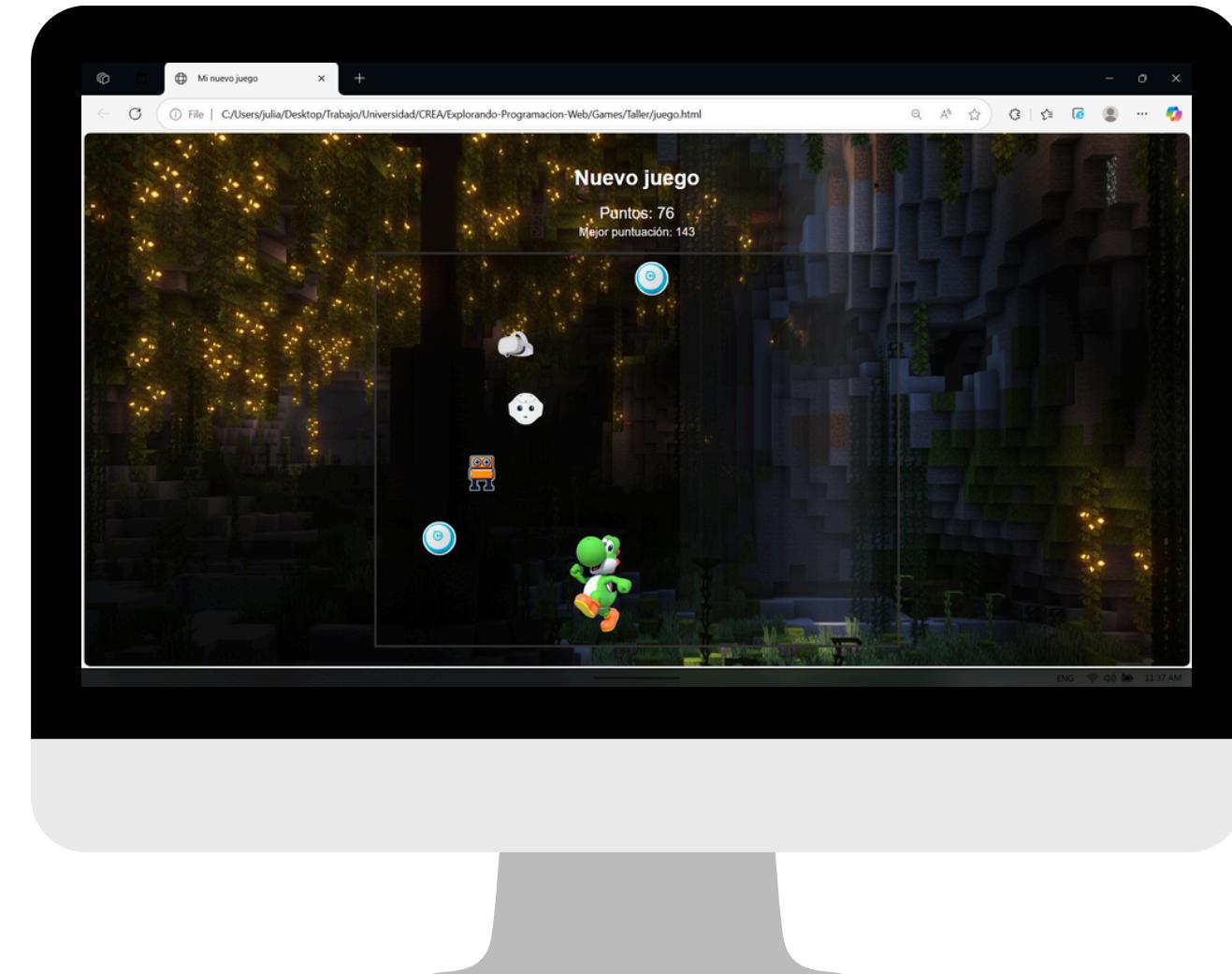
juego.js

```

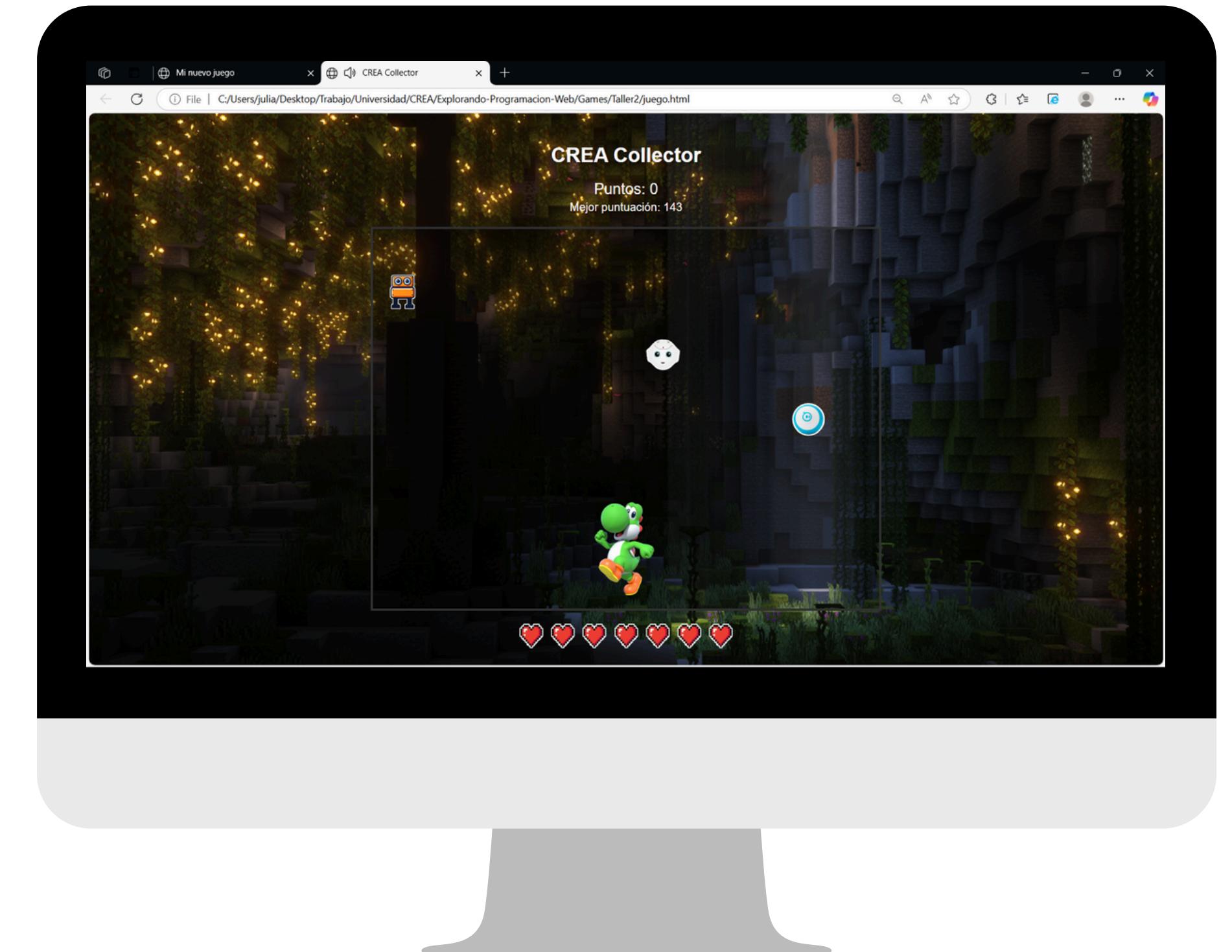
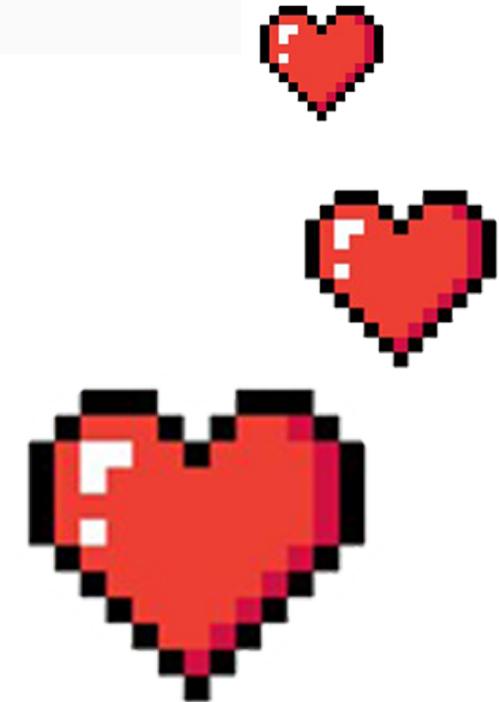
// Cada objeto debe tener: tipo, clase, puntos, y probabilidad
// La suma de todas las probabilidades debe ser menor o igual a 1
OBJETOS = [
{
  tipo: 'bot',          // Nombre del objeto
  clase: 'objeto-bot', // Clase CSS para dar estilo
  puntos: 1,            // Puntos que da al recogerlo
  probabilidad: 0.25   // Probabilidad de que aparezca (entre 0 y 1)
},
// Tarea 4: Agrega 3 objetos que den puntos positivos
{
  tipo: 'sphero',        // Nombre del objeto
  clase: 'objeto-sphero', // Clase CSS para dar estilo
  puntos: 2,              // Puntos que da al recogerlo
  probabilidad: 0.20    // Probabilidad de que aparezca (entre 0 y 1)
},
{
  tipo: 'realidad virtual',      // Nombre del objeto
  clase: 'objeto-vr', // Clase CSS para dar estilo
  puntos: 5,            // Puntos que da al recogerlo
  probabilidad: 0.15   // Probabilidad de que aparezca (entre 0 y 1)
},
{
  tipo: 'nova',          // Nombre del objeto
  clase: 'objeto-nova', // Clase CSS para dar estilo
  puntos: 10,             // Puntos que da al recogerlo
  probabilidad: 0.10   // Probabilidad de que aparezca (entre 0 y 1)
}
];

```

# Possible solution



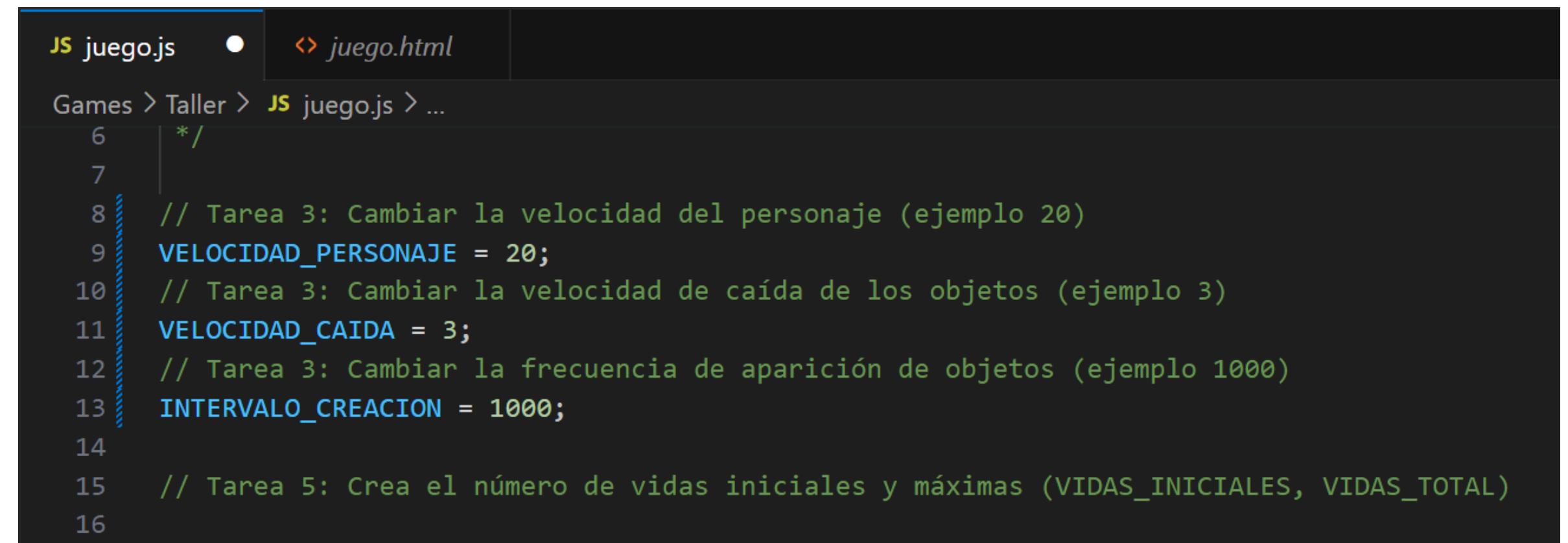
# Vidas



# Tarea 5:

## juego.js

Definamos la cantidad de **vidas** **iniciales** y **vidas en total** que va a tener nuestro personaje



The screenshot shows a code editor window with the file "juego.js" open. The file path in the title bar is "Games > Taller > juego.js". The code itself is as follows:

```
JS juego.js ● ⚡ juego.html
Games > Taller > JS juego.js > ...
6  /*
7
8 // Tarea 3: Cambiar la velocidad del personaje (ejemplo 20)
9 VELOCIDAD_PERSONAJE = 20;
10 // Tarea 3: Cambiar la velocidad de caída de los objetos (ejemplo 3)
11 VELOCIDAD_CAIADA = 3;
12 // Tarea 3: Cambiar la frecuencia de aparición de objetos (ejemplo 1000)
13 INTERVALO_CREACION = 1000;
14
15 // Tarea 5: Crea el número de vidas iniciales y máximas (VIDAS_INICIALES, VIDAS_TOTAL)
16
```



## Solución

### Tarea 5:

**Se crean las variables VIDAS\_INICIALES y VIDAS\_TOTAL para guardar el número de vidas.**

### juego.js

```
15 // Tarea 5: Crea el número de  
16 VIDAS_INICIALES = 7;  
17 VIDAS_TOTAL = 7;  
18
```



# Tarea 6: Completa la función

## juego.js

**Hay que completar  
la función que crea  
los corazones  
iniciales.**

```
JS juego.js ● ⚡ juego.html
Games > Taller > JS juego.js > actualizarVidasVisualmente
49 // Tarea 6: Completa la función para crear los corazones
50 function crearCorazones() {
51     /*
52     const contenedor = document.getElementById('hearts-container');
53
54     contenedor.innerHTML = '';
55
56     // Aquí crearemos tantos corazones como VIDAS INICIALES hayamos definido
57     for (let i = 0; i < VIDAS; i++) { // Tarea 6: Reemplaza VIDAS por la variable creada
58         const corazon = document.createElement('img');
59         corazon.src = ''; // Tarea 6: Agrega la imagen del corazón rojo
60         corazon.classList.add('heart');
61         contenedor.appendChild(corazon);
62     }
63     */
64 }
```

**La función está comentada. Para “activarla” deberás borrar las líneas 51 y 63.**



# Vidas

Se reemplaza VIDAS  
por  
**VIDAS\_INICIALES** y  
se agrega la imagen  
de un **corazón rojo** a  
las vidas.

## Solución

### juego.js

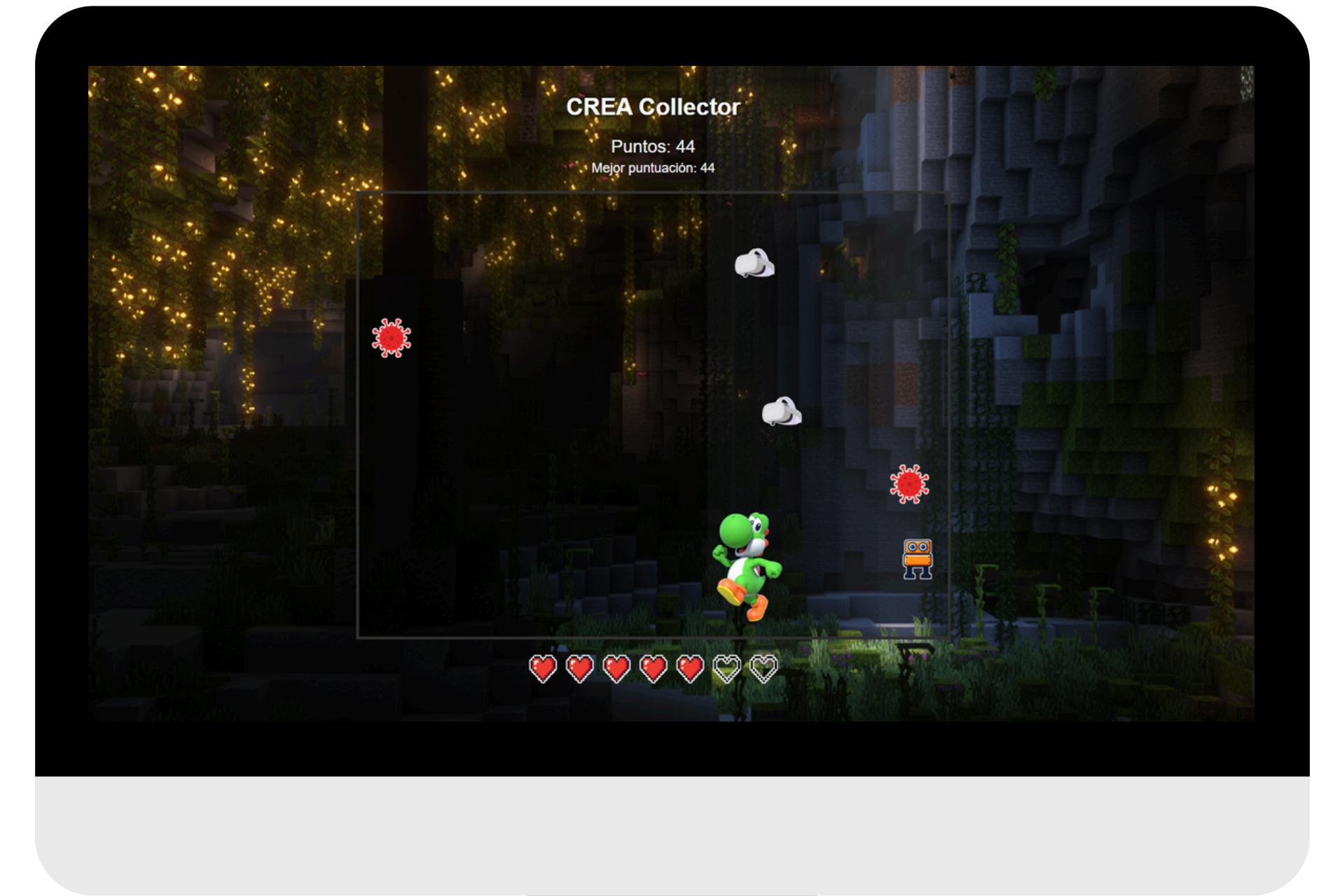
```
// Tarea 6: Completa la función para crear los corazones
function crearCorazones() {
    const contenedor = document.getElementById('hearts-container');

    contenedor.innerHTML = '';

    // Aquí crearemos tantos corazones como VIDAS_INICIALES hayamos definido
    for (let i = 0; i < VIDAS_INICIALES; i++) { // Tarea 6: Reemplaza VIDAS por la variable creada
        const corazon = document.createElement('img');
        corazon.src = '../../../../../Assets/Taller/Uniandes_Heart.png'; // Tarea 6: Agrega la imagen del corazón rojo
        corazon.classList.add('heart');
        contenedor.appendChild(corazon);
    }
}
```

La imagen del corazón también se puede cambiar si se desea. Pero esto tiene una implicación adicional que explicaremos más adelante.

# Perder Vidas



# Tarea 7

## juego.js

**Cambia uno de los objetos para que le quite vidas a nuestro personaje.**

**Ayuda: Los puntos negativos quitan vidas**

```
20  OBJETOS = [
21  {
22    tipo: 'bot',           // Nombre del objeto
23    clase: 'objeto-bot',  // Clase CSS para dar estilo
24    puntos: 1,             // Puntos que da al recogerlo
25    probabilidad: 0.25   // Probabilidad de que aparezca (entre 0 y 1)
26  },
27  // Tarea 4: Agrega 3 objetos que den puntos positivos
28  {
29    tipo: 'sphero',        // Nombre del objeto
30    clase: 'objeto-sphero', // Clase CSS para dar estilo
31    puntos: 2,             // Puntos que da al recogerlo
32    probabilidad: 0.20   // Probabilidad de que aparezca (entre 0 y 1)
33  },
34  {
35    tipo: 'realidad virtual', // Nombre del objeto
36    clase: 'objeto-vr',    // Clase CSS para dar estilo
37    puntos: 5,              // Puntos que da al recogerlo
38    probabilidad: 0.15   // Probabilidad de que aparezca (entre 0 y 1)
39  },
40  {
41    tipo: 'nova',          // Nombre del objeto
42    clase: 'objeto-nova',  // Clase CSS para dar estilo
43    puntos: 10,             // Puntos que da al recogerlo
44    probabilidad: 0.10   // Probabilidad de que aparezca (entre 0 y 1)
45  }
46  // Tarea 8: Agrega 1 objeto que quite vidas
47  // Ayuda: Los puntos negativos quitan vidas
48];
49
50
```

# Perder Vidas

En este ejemplo creamos un objeto que **quita 1 vida**. Es de **tipo virus** y su **clase** es '**objeto-virus**'.

## Possible solución juego.js

```
39  ],
40  {
41    tipo: 'nova',           // Nombre del objeto
42    clase: 'objeto-nova',  // Clase CSS para dar estilo
43    puntos: 10,            // Puntos que da al recogerlo
44    probabilidad: 0.10    // Probabilidad de que aparezca (entre 0 y 1)
45  },
46  {
47    tipo: 'virus',          // Nombre del objeto
48    clase: 'objeto-virus', // Clase CSS para dar estilo
49    puntos: -1,             // Puntos que da al recogerlo
50    probabilidad: 0.15    // Probabilidad de que aparezca (entre 0 y 1)
51  }
52 }
53 ];
54 ];
```

¡Pueden hacer que cualquier objeto quite vidas!

# Perder Vidas

juego.js

**Hay que completar la función que actualiza las vidas usando nuestras imágenes.**

**Esta función se encarga de cambiar la imagen de los corazones cuando el personaje pierde una vida.**

**Por esta razón se recomienda usar corazones rojos y vacíos. Pero pueden usar otras imágenes.**



```

JS juego.js • juego.html •
Games > Taller > JS juego.js > ...
68 function actualizarVidasVisualmente() {
69     /*
70      console.log("Actualizando vidas visualmente:", vidas);
71
72      // Obtener el contenedor de corazones
73      const heartsContainer = document.getElementById('hearts-container');
74      if (!heartsContainer) {
75          console.error("Error: Contenedor de corazones no encontrado");
76          return;
77      }
78
79      // Obtener todas las imágenes de corazones
80      const corazones = heartsContainer.querySelectorAll('.heart');
81
82      console.log("Corazones encontrados:", corazones.length);
83
84      // Si no hay corazones, crearlos de nuevo
85      if (corazones.length === 0) {
86          console.log("No hay corazones, creándolos de nuevo");
87          crearCorazones();
88          return;
89      }
90
91      // Actualizar cada corazón
92      // Tarea 10: Completa la función
93      for (let i = 0; i < corazones.length; i++) {
94          if (i < vidas) {
95              corazones[i].src = ''; // Tarea 10: Imagen para vida llena
96          } else {
97              corazones[i].src = ''; // Tarea 10: Imagen para vida vacía
98          }
99
100         // Asegurarse de que el corazón sea visible
101         corazones[i].style.display = 'inline-block';
102     }
103     */
104 }

```

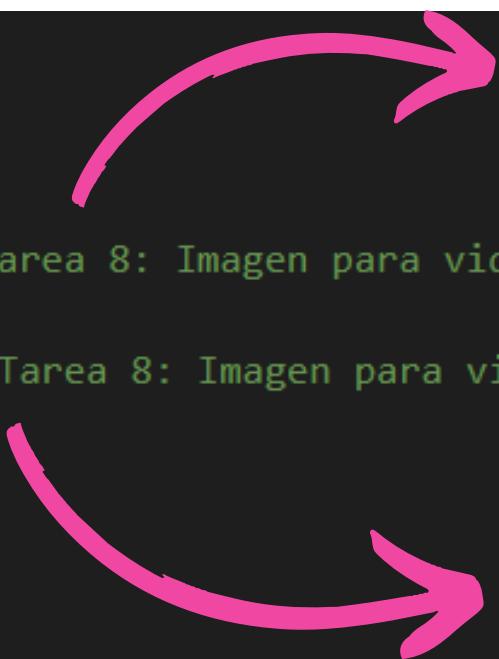
**La función está comentada. Para “activarla” deberás borrar las líneas 69 y 103.**

# Tarea 8: Completa la función

## Solución

### juego.js

```
91 // Actualizar cada corazón
92 // Tarea 8: Completa la función
93 for (let i = 0; i < corazones.length; i++) {
94     if (i < vidas) {
95         corazones[i].src = '../../../../../Assets/CorazónRojo.png'; // Tarea 8: Imagen para vida llena
96     } else {
97         corazones[i].src = '../../../../../Assets/CorazónVacio.png'; // Tarea 8: Imagen para vida vacía
98     }
99
100    // Asegurarse de que el corazón sea visible
101    corazones[i].style.display = 'inline-block';
102}
103}
104}
```



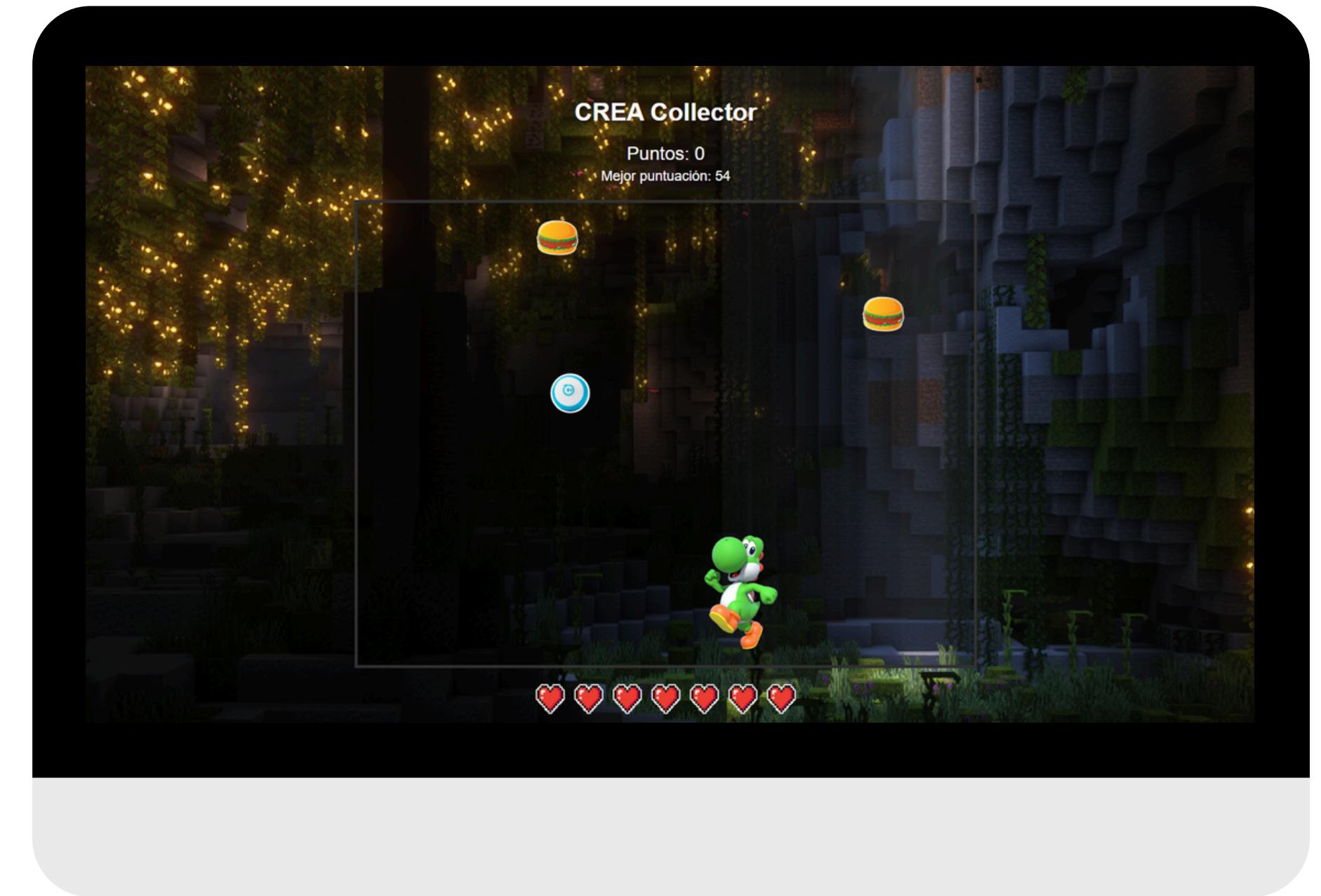
**La primera línea  
representa el corazón  
completo.**



**La segunda línea  
representa el  
corazón después de  
perder una vida.**



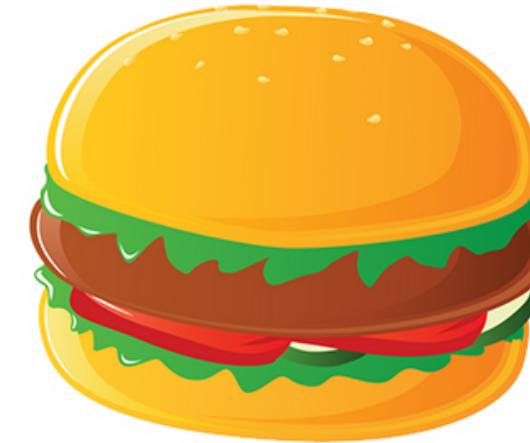
# Ganar Vidas



# Tarea 9

**Edita un objeto para que le devuelva vidas a nuestro personaje.**

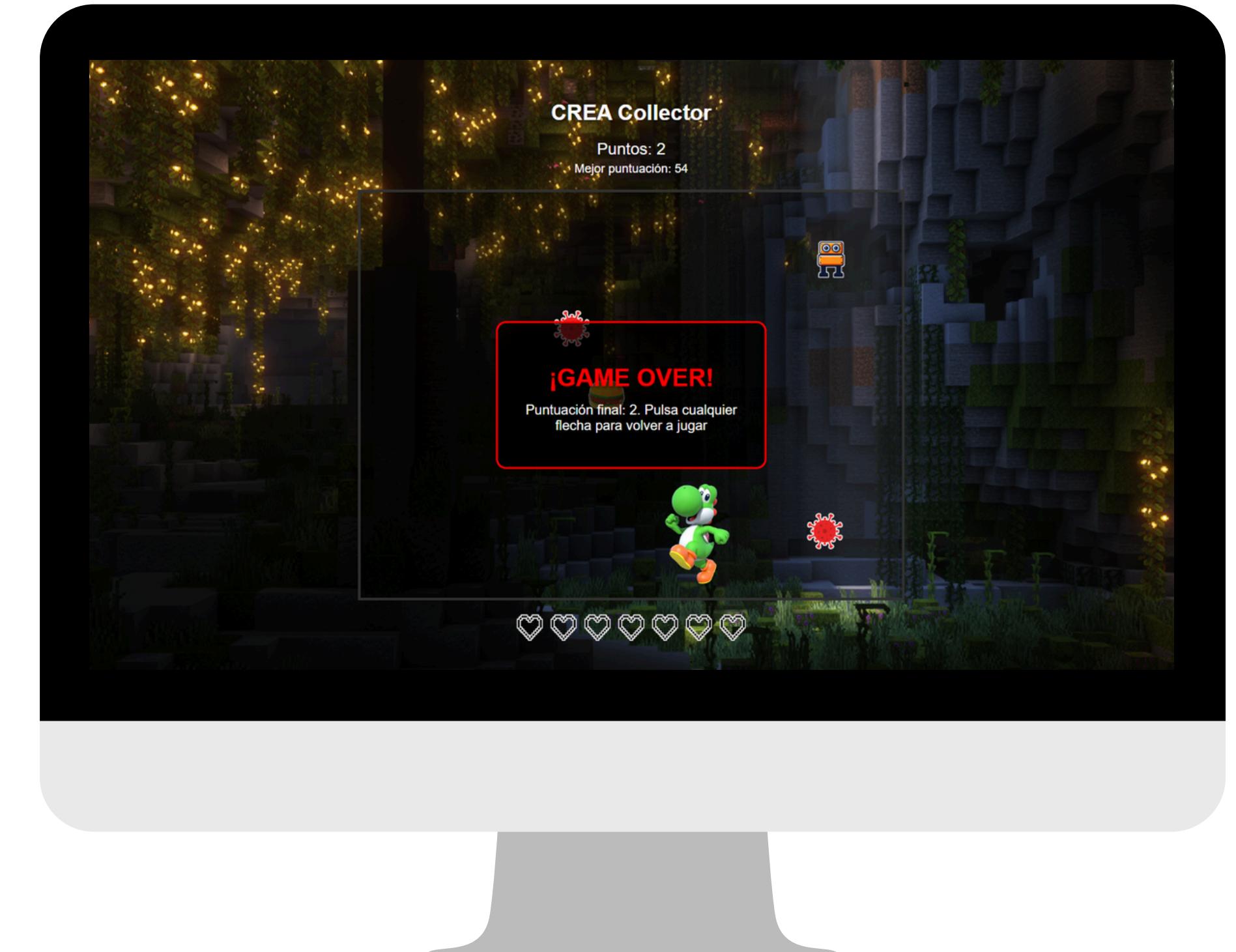
**Ayuda:** ¿Debería dar puntos?  
¿Podemos agregar algo extra?



```
{  
    tipo: 'hamburguesa',           // Nombre del objeto  
    clase: 'objeto-hamburguesa',  // Clase CSS para dar estilo  
    puntos: 0,                     // Puntos que da al recogerlo  
    vidaExtra: true,               // Indica que da vida extra  
    probabilidad: 0.10            // Probabilidad de que aparezca (entre 0 )  
};
```

**Usamos un nuevo atributo llamado **vidaExtra** el cual es un **booleano**.**

# Game Over



# Tarea 10

## juego.js

**Edita un objeto que le  
ocasione un Game Over.**

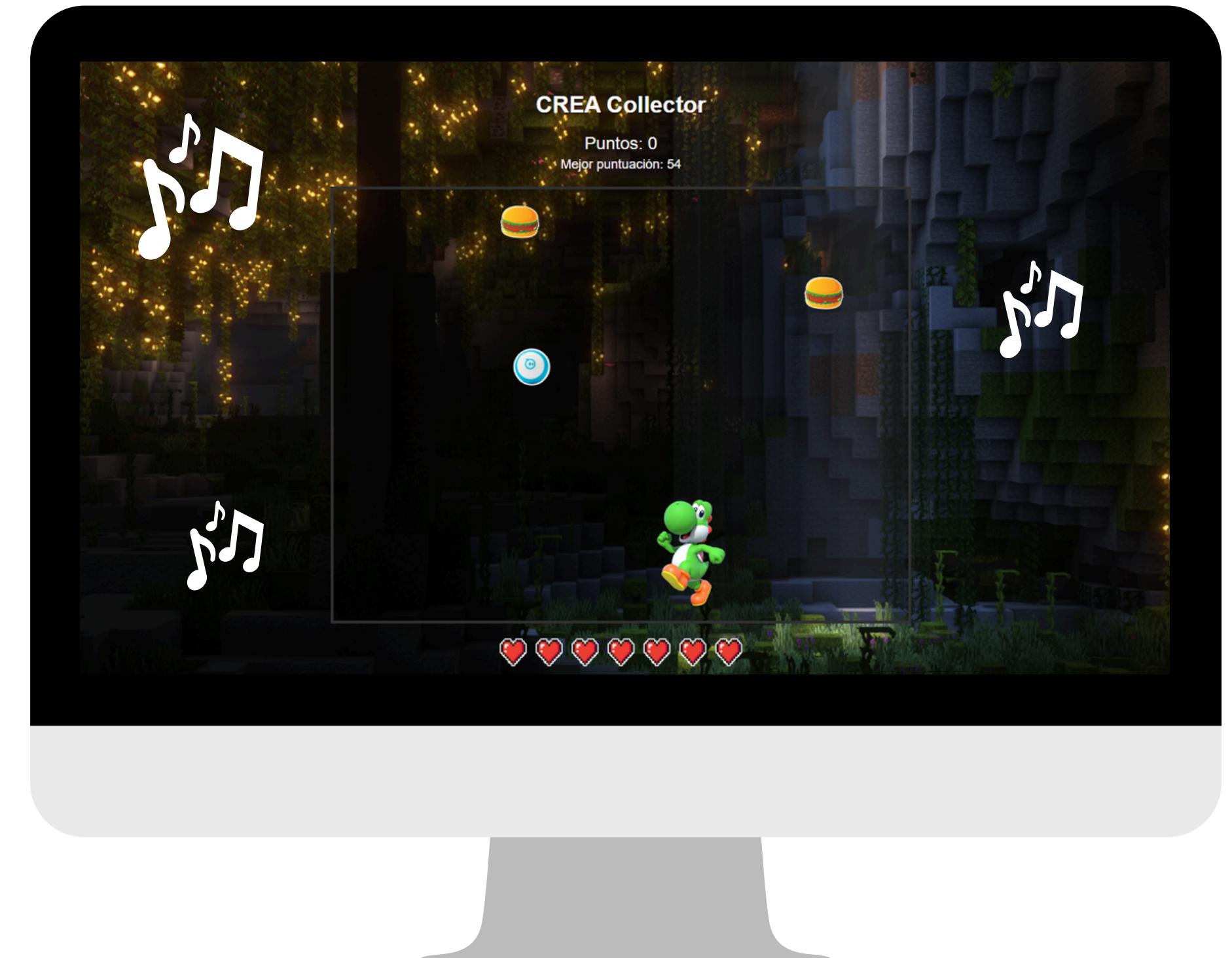
```
65      {  
66          tipo: 'vinchi',           // Nombre del objeto  
67          clase: 'objeto-vinchi',   // Clase CSS para dar estilo  
68          puntos: 0,              // Puntos que da al recogerlo  
69          gameOver: true,         // Indica que termina el juego  
70          probabilidad: 0.15     // Probabilidad de que aparezca (entre 0 y 1)  
71      }  
72  ];
```

**Ayuda:** Recuerda lo hecho  
en el paso anterior  
(gameOver).

**Usamos un nuevo atributo  
llamado **gameOver** el cual  
es un **booleano**.**



# Sonido



# Sonido

## juego.html

**Revisemos los audios incluidos en el esqueleto de la página.**

**¿Qué nombres (**id**) tienen?**

```
54      <!-- Sonidos del juego -->
55      <audio id="sonidoPunto" src="../../Assets/Punto.wav" preload="auto"></audio>
56      <audio id="sonidoPerdida" src="../../Assets/Perdida.wav" preload="auto"></audio>
57      <audio id="sonidoGameOver" src="../../Assets/GameOver.wav" preload="auto"></audio>
58
59
60      <!-- Conexión al archivo de lógica, luego el de juego -->
61      <!-- Tarea 3: Conecta tus archivos JavaScript -->
62      <script src="logica.js"></script>
63      <script src="juego.js"></script>
64  </body>
65  </html>
```



# Tarea 11

## juego.js

**Teniendo en cuenta los nombres anteriores, completa la función para reproducir un sonido.**

```
JS juego.js • ⌂ juego.html • # juego.css •
Games > Taller > JS juego.js > document.addEventListener('DOMContentLoaded') callback
105
106 // Tarea 11: Completa la función para reproducir sonidos
107 function reproducirSonido(tipo) {
108     /*
109     switch(tipo) {
110         case 'punto':
111             document.getElementById('').currentTime = 0;
112             document.getElementById('').play();
113             break;
114         case 'perdida':
115             document.getElementById('').currentTime = 0;
116             document.getElementById('').play();
117             break;
118         case 'gameover':
119             document.getElementById('').currentTime = 0;
120             document.getElementById('').play();
121             break;
122     }
123     */
124 }
```

**¿Qué significan los `case`?**

La función está comentada. Para “activarla” deberás borrar las líneas 108 y 121.

# Sonido

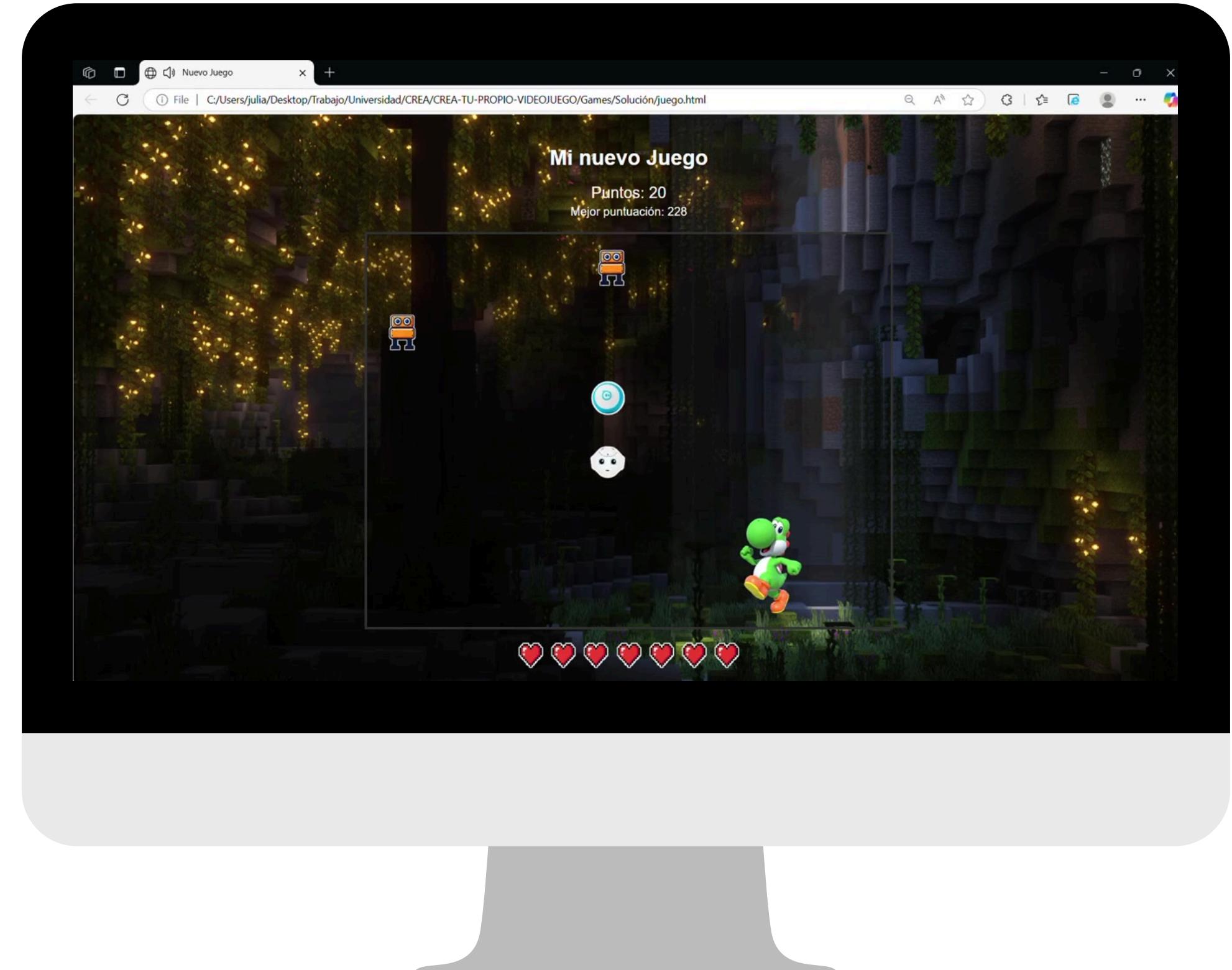
## Possible solución juego.js

Para cada uno de los casos, activamos un sonido diferente



```
JS juego.js ● ⚡ juego.html ● # juego.css ●
Games > Taller > JS juego.js > window.addEventListener('load') callback
105
106 // Tarea 11: Completa la función para reproducir sonidos
107 function reproducirSonido(tipo) {
108
109     switch(tipo) {
110         case 'punto':
111             document.getElementById('sonidoPunto').currentTime = 0;
112             document.getElementById('sonidoPunto').play();
113             break;
114         case 'perdida':
115             document.getElementById('sonidoPerdida').currentTime = 0;
116             document.getElementById('sonidoPerdida').play();
117             break;
118         case 'gameover':
119             document.getElementById('sonidoGameOver').currentTime = 0;
120             document.getElementById('sonidoGameOver').play();
121             break;
122     }
123 }
```

# Resultado final

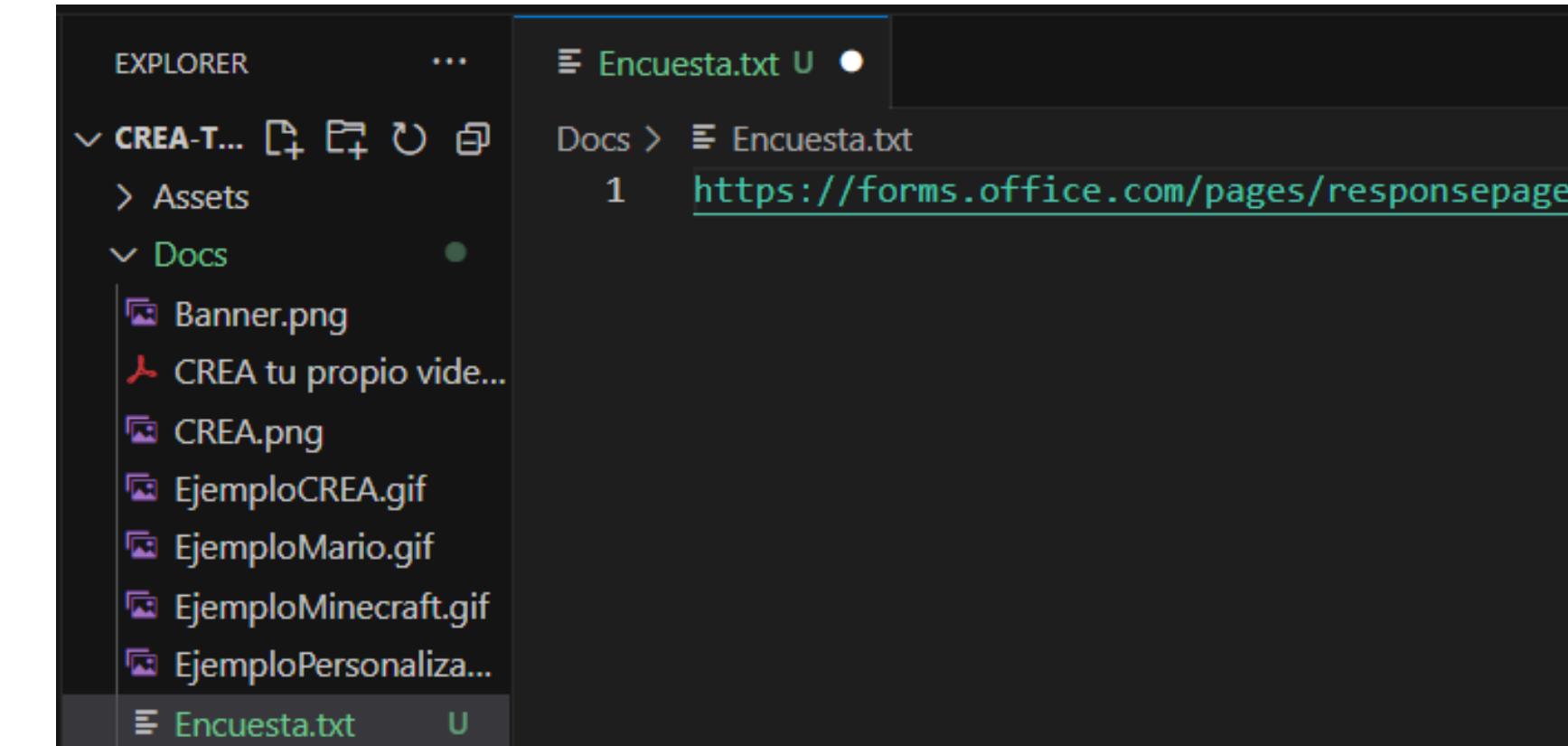


# ¡Felicidades!

Hoy no solo jugaron un videojuego, ¡lo **crearon!** Imaginen todo lo que pueden lograr si siguen **aprendiendo**.



# Califica tu experiencia



Encuentras el enlace en:  
**Docs/Encuesta.txt**

