2020

Juego 2D desarrollado en JavaScript puro: Introducción a la Informática

HEMERSON CARDONA CARDONA

OCTUBRE DE 2020

UTP | Pereira

1 CONTENIDO

2 PRESENTACIÓN

La presente monografía describe el desarrollo metódico de un juego 2D elaborado utilizando HTML5, CSS, CANVAS y JavaScript.

El juego elaborado se crea con JavaScript puro, utilizando un enfoque metódico en el cual se avanza de versión en versión, de modo que cada nuevo programa abarca un aspecto adicional del juego.

Cada una de las fases se cubre en un apartado diferente. Se plantea el alcance de cada una de ellas, se explican las instrucciones o conceptos que son necesarios para entender el significado del trabajo realizado, se agrega el código, y finalmente se presentan las gráficas de la ejecución del programa.

Una vez cubiertas todas las fases, se dispondrá de un clásico juego 2D que servirá como base e inspiración para desarrollar otros programas aplicados en la Web.

Gráfica 1. Juego 2D en JavaScript.

El documento web que sirve como referencia para el desarrollo del juego está en el siguiente enlace:

https://developer.mozilla.org/es/docs/Games/Workflows/Famoso_juego_2D_usando_JavaScript_puro

AUTOR: HEMERSON CARDONA CARDONA

3 FASE 1: DIBUJAR Y MOVER UNA BOLA

El primer paso consiste en elaborar una página HTML básica. Agregaremos a dicha página un elemento CANVAS, el cual nos servirá como base para el desarrollo del juego 2D.

El código JavaScript que operará sobre el CANVAS debe encerrarse entre las etiquetas `<script>…</script>`

La correcta visualización del CANVAS requiere de la adición de algunas características de estilo. Una vez hecho esto, se procede a establecer la codificación pertinente del JavaScript. Debe notarse la inclusión de algunas variables que definen la funcionalidad del juego en sus aspectos básicos: las coordenadas en las que se encuentra la bola y los valores de incremento para modificar su posición.

Se definen tres funciones importantes. La primera de ellas, dibujarBola(), se encarga de dibujar sobre la pantalla una bola con el color indicado en los estilos. La segunda función se denomina dibujar(), y es la encargada de limpiar el CANVAS, dibujar la bola y cambiar los valores de las coordenadas. Finalmente, la función setInterval(dibujar, 10), llama a la función dibujar cada 10 milisegundos.

El código fuente del programa es el siguiente (para darle formato, se deben seguir las instrucciones disponibles en: https://trabajonomada.com/insertar-codigo-word/ y seguidamente utilizar el enlace: http://qbnz.com/highlighter/php_highlighter.php)

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D: JavaScript - 01</title>
6.
7.     <!-- Define los estilos de la interfaz visual
8.          padding es la distancia de un objeto en relación con el
            marco que lo contiene
9.          margin es la distancia que separa a un objeto de otro
10.              background es el color de fondo
11.              display: block; Estos elementos fluyen hacia abajo
12.              margin: 0 auto; Centra el canvas en la pantalla -->
13.         <style>
14.             * {
15.                 padding: 0;
16.                 margin: 0;
17.             }
18.             canvas {
19.                 background: #eee;
20.                display: block;
21.                 margin: 0 auto;
22.             }
23.         </style>
24.
25.     </head>
26.     <body>
27.
28.     <canvas id="miCanvas" width="480" height="320"></canvas>
29.
30.     <script>
31.         var canvas = document.getElementById("miCanvas");
32.         var ctx = canvas.getContext("2d");
33.
34.         // Coloca x en la mitad del ancho deL CANVAS
35.         var x = canvas.width/2;
36.
37.         // Coloca y en la mitad de la altura del CANVAS (restando
               30 a dicho valor)
38.         var y = canvas.height-30;
39.
40.         /* DEFINE LOS INCREMENTOS EN X y en Y. El valor dy es
               negativo
41.            para que inicialmente el movimiento de la bola sea
               hacia arriba */
42.         var dx = 7;
43.         var dy = -7;
44.
45.         function dibujarBola() {
46.             // Inicia el dibujo
47.             ctx.beginPath();
48.
49.             /* Define un círculo en las coordenadas (x, y) con
```

```
50.                 radio 10
51.                 El ángulo va desde 0 hasta 2*PI (360 grados) */
52.             ctx.arc(x, y, 10, 0, Math.PI*2);
53.
54.             // Color de llenado
55.             ctx.fillStyle = "#0095DD";
56.
57.             // Se llena el círculo con el color indicado
58.             ctx.fill();
59.
60.             // Finaliza el dibujo
61.             ctx.closePath();
62.         }
63.
64.         /* LA FUNCIÓN dibujar REALIZA TRES TAREAS:
65.            1) Limpia el CANVAS. Inicio= (0,0) Ancho=canvas.width
                  Altura=canvas.height
66.            2) Dibuja una bola en las coordenadas (x, y)
67.            3) Cambiar las coordenadas (x, y) agregando los
                  valores dx, dy
68.               Con este cambio cada vez que se dibuja la bola,
                  está en una nueva posición */
69.         function dibujar() {
70.
71.             // Limpia el CANVAS
72.             ctx.clearRect(0, 0, canvas.width, canvas.height);
73.
74.             // Dibuja la bola
75.             dibujarBola();
76.
77.             // Se incrementa x en el valor dx
78.             x = x + dx;
79.
80.             // Se incrementa y en el valor dy
81.             y = y + dy;
82.         }
83.
84.         /* EJECUTA LA FUNCIÓN dibujar CADA 10 MILISEGUNDOS
85.            Este es el mecanismo utilizado para construir un
               sistema que
86.            ejecuta acciones de manera permanente y periódica */
87.         setInterval(dibujar, 10);
88.     </script>
89.
90.     </body>
91.     </html>
```

Gráfica 2. El código del juego. Fase 1.

Al ejecutar este código se obtiene la siguiente interfaz visual:

Gráfica 3. La interfaz inicial del juego.

En la gráfica 3 se aprecia el dibujo de la bola, y la secuencia de movimiento a partir de los incrementos en X y Y que fueron definidos.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

4 FASE 2: REBOTANDO EN LAS PAREDES.

El segundo paso consiste en elaborar los límites permitidos a los que la bola puede llegar y en los que rebotara y así mantenerse dentro del cuadro asignado.

En este paso se crean dos condiciones las cuales generan los límites permitidos a los que la bola puede llegar y va a rebotar:

La primera condición es if(x + dx > canvas.width-ballRadius || x + dx < ballRadius) { dx = -dx;}, esta condición crea el rango horizontal al que la pelota se puede desplazar.

La segunda condición es if(y + dy > canvas.height-ballRadius || y + dy < ballRadius) {dy = -dy;}, esta condición crea el rango vertical al que la pelota se puede desplazar.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - lección 02</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.     <canvas id="miCanvas" width="480" height="320"></canvas>
11.
12.     <script>
13.         var canvas = document.getElementById("miCanvas");
14.         var ctx = canvas.getContext("2d");
15.         var ballRadius = 10;
16.         var x = canvas.width/2;
17.         var y = canvas.height-30;
18.         var dx = 2;
19.         var dy = -2;
20.
21.         function dibujarBola() {
22.             ctx.beginPath();
23.             ctx.arc(x, y, ballRadius, 0, Math.PI*2);
24.             ctx.fillStyle = "#0095DD";
25.             ctx.fill();
26.             ctx.closePath();
27.         }
28.
29.         function dibujar() {
30.             ctx.clearRect(0, 0, canvas.width, canvas.height);
31.             dibujarBola();
```

```
32.
33.              /* IMPORTANTE:
34.
35.                 EL OPERADOR || es el operador lógico OR
36.                 Este operador se utiliza para indicar la condición
                    de conjunción
37.                 SI SE CUMPLE UNA CONDICIÓN, O SE CUMPLE OTRA
                    CONDICIÓN, ENTONCES
38.                 SE CUMPLE LA CONDICIÓN
39.
40.                 EL OPERADOR && es el oerador lógico AND
41.                 Este operador se utiliza para indicar la condicion
                    de disyunción
42.                 SI SE CUMPLE UNA CONDICIÓN, Y SE CUMPLE OTRA
                    CONDICIÓN (simultánea), ENTONCES
43.                 SE CUMPLE LA CONDICIÓN
44.
45.             */
46.
47.             /* DESPUÉS DE DIBUJAR LA BOLA, SE DEBEN CAMBIAR LAS
                   COORDENADAS
48.                EN LA lección 01 NO SE TENÍA CONTROL SOBRE LOS
                   LÍMITES DE LA CAJA
49.                ------------------------------------------------
   --------------
50.                SI x + dx ES MAYOR AL ANCHO DEL CANVAS O MENOR AL
                   TAMAÑO DEL
51.                RADIO DE LA BOLA (caso en el cual se encuentra
                   hacia la izquierda)
52.                SE CAMBIA LA DIRECCIÓN DE AVANCE HORIZONTAL.
53.                ESTO SE LOGRA CAMBIANDO EL SIGNO DE LA VARIABLE dx
54.                ESTO HACE QUE SE CAMBIE EL SENTIDO DEL MOVIMIENTO
                   HORIZONTAL */
55.             if(x + dx > canvas.width-ballRadius || x + dx <
   ballRadius) {
56.                 dx = -dx;
57.              }
58.
59.              /* SI y + dy ES MAYOR A LA ALTURA DEL CANVAS O MENOR
   AL TAMAÑO DEL
60.                 RADIO DE LA BOLA, SE CAMBIA LA DIRECCIÓN DEL
   AVANCE VERTICAL.
61.                 ESTO SE LOGRA CAMBIANDO EL SIGNO DE LA VARIABLE dy
62.                 ESTE CAMBIO EN dy HACE QUE SE MUEVA VERTICALMENTE
   EN SENTIDO
63.                 OPUESTO */
64.              if(y + dy > canvas.height-ballRadius || y + dy <
   ballRadius) {
65.                 dy = -dy;
66.             }
67.
68.             /* AQUÍ SE CAMBIA LA POSICIÓN DE LA BOLA. SE TOMA EN
   CUENTA LAS
```

```
69.                 MODIFICACIONES A dx y dy, EN CASO DE QUE SE
   HUBIERAN PRODUCIDO */
70.             x += dx;
71.             y += dy;
72.         }
73.
74.         setInterval(dibujar, 10);
75.     </script>
76.
77.     </body>
78.     </html>
```

Gráfica 4. Código para límites.

Al ejecutar este código se obtiene la siguiente interfaz visual:

Gráfica 5. Ejecución de los límites.

En la gráfica 5 podemos observar a la bola rebotando y cumpliendo con los límites anteriormente definidos en las condiciones.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

5 FASE 3: CONTROL DE LA PALETA Y EL TECLADO

El paso numero 3 consiste en crear la paleta en la cual rebotará la bola, la cual estará situada en la parte inferior del juego y será controlada por ambas flechas del teclado.

Para empezar con el tercer paso primero se deben crear dos variables a las cuales se les asignara el movimiento de la paleta con las flechas del teclado. Estas variables llevaran el siguiente nombre flechaDerechaPulsada y flechaIzquierdaPulsada. Luego de esto se crea una función la cual maneja el movimiento de la tecla presionada y otro de la tecla liberada.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #03 - Paleta y Control por Teclado</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.      <canvas id="miCanvas" width="480" height="320"></canvas>
11.
12.      <script>
13.          var canvas = document.getElementById("miCanvas");
14.          var ctx = canvas.getContext("2d");
15.
16.          /* Variables básicas:
17.
18.             radioBola: radio de la esfera
19.             x: columna en la que se encuentra situada la bola
20.             y: fila en la que se encuentra situada la bola
21.             dx: desplazamiento horizontal de la bola
22.             dy: desplazamiento vertical de la bola
23.
24.             NOTAS: originalmente, la bola está en centro del
   CANVAS
25.                    en el sentido horizontal. Y se encuentra en la
26.                    base inferior, pues el eje Y crece de arriba
   hacia
27.                    abajo. A este valor se le resta 30, para tomar
   en
28.                    cuenta el tamaño de la bola (que es de 20 si
   tomamos
```

```
29.                     en cuenta el diámetro)
30.
31.             NOTAS: El desplazamiento en el eje X y en el eje Y,
   son
32.                     controlados por la variable dx y la variable
   dy.
33.                     Estos valores son de 2 pixeles, y gracias a
   este
34.                     avance que se realiza en un ciclo ejecutado
   cada
35.                     10 milisegundos, se genera el efecto de avance
   de
36.                     la bola. Dentro del ciclo se cambia la
   coordenada
37.                     (x, y) agregando los valores (dx, dy), motivo
   por
38.                     el cual la bola cambia su posición cada 10
   milisegundos
39.           */
40.           var radioBola = 10;
41.           var x = canvas.width/2;
42.           var y = canvas.height-30;
43.           var dx = 2;
44.           var dy = -2;
45.
46.           /* Las variables a continuación, tienen el siguiente
   significado:
47.
48.                 Se define una paleta en la que rebotará la bola
49.                 La paleta está situada en la base de la pantalla
   de juego
50.                 Dicha paleta será controlada por la flecha
   izquierda y
51.                 la flecha derecha del teclado (luego será
   controlador por el ratón)
52.
53.                 alturaPaleta: define la altura de la paleta en
   pixeles
54.                 anchuraPaleta: define la anchura de la paleta
55.
56.                 NOTA: Estos dos valores determinan el tamaño de la
   paleta
57.                       La paleta se encuentra situada en la base de
   la pantalla
58.                       Para calcular la posición en X de la paleta,
   se debe tomar
59.                       el ancho del CANVAS, restarle la anchura de
   la paleta, y
60.                       el espacio que sobre debe dividirse entre
   dos
61.                       Esto garantiza que originalmente la paleta
   estará centrada
62.                       en la base de la pantalla
```

```
63.
64.                 Al inicio del juego, aún no se ha presionado
   ninguna de las
65.                 flechas. Esta es la razón por la cual se definen
   dos variables que
66.                 "recuerdan" cual de las flechas se ha presionado,
   pero que
67.                 inicialmente están puestas a: false, indicando el
   estado inicial
68.                 Cuando se pulse cualquiera de las dos flechas, su
   valor será:
69.                 true (verdadero), y este valor permitira
   establecer en qué
70.                 dirección se debe mover la paleta (dentro del
   ciclo del juego)
71.                 Las variables son:
72.
73.                 flechaDerechaPulsada
74.                 flechaIzquierdaPulsada
75.
76.                 NOTA: Desde ahora debe tomarse en cuenta que
77.                 cuando se pulse
78.                 cualquiera de las dos flechas, solamente se hará
79.                 un
80.                 desplazamiento de la paleta a la izquierda o hacia
81.                 La derecha. Si se mantiene pulsada la tecla, la
   paleta se
82.                       continuará desplazando, hasta alcanzar el
   extremo derecho
83.                       o izquierdo de la pantalla del juego
84.          */
85.          var alturaPaleta = 10;
86.          var anchuraPaleta = 75;
87.          var paletaPosX = (canvas.width-anchuraPaleta)/2;
88.          var flechaDerechaPulsada = false;
89.          var flechaIzquierdaPulsada = false;
90.
91.          /* La instruccion: addEventListener, se utiliza para
   crear un
92.             mecanismo de respuesta ante eventos que se produzcan
   en el juego
93.
94.             addEventListener "agregar un mecanismo que deteca y
   recibe eventos"
95.
96.             addEventListener recibe tres parámetros:
97.
98.             1) El evento que se va a detectar
99.             2) El nombre que le asignamos a la función que
   responde ante el evento
100.            3) Valor true o false que determina la reacción ante
   el evento
101.
```

```
102.            Los dos primeros parámetros son fáciles de entender.
   Pero el tercero
103.            requiere de una explicación adicional:
104.
105.            Para entender el tercer parámetro, primero hemos de
   saber lo que es
106.            el flujo de eventos.
107.
108.                Supongamos que tenemos este tres objetos en la
   página:
109.
110.            <body>
111.                <div>
112.                    <button>HAZME CLIC</button>
113.                </div>
114.            </body>
115.
116.            El <body> contiene un <div>, y dentro de él esta
   un <button>
117.
118.            Cuando hacemos clic en el botón no sólo lo estamos
   haciendo sobre él,
119.            sino sobre los elementos que lo contienen en el árbol
   de la página,
120.            es decir, hemos hecho clic, además, sobre el
   elemento <body> y sobre
121.            el elemento <div>. Sí sólo hay una función asignada a
   una escucha
122.            para el botón, no hay mayor problema, pero si además
   hay una
123.            eschucha para el body y otra para el div,
124.            ¿cuál es el orden en que se deben lanzar las tres
   funciones?
125.
126.                Para contestar a esa pregunta existe un modelo de
   comportamiento,
127.                el flujo de eventos. Según éste, cuando se hace
   clic sobre un
128.                elemento, el evento se propaga en dos fases, una
   que es la
129.                captura —el evento comienza en el nivel superior
   del documento
130.                y recorre los elementos de padres a hijos— y la
   otra la burbuja
131.                —el orden inverso, ascendiendo de hijos a padres—.
132.
133.                Así, el orden por defecto de lanzamiento de las
   funciones
134.                de escucha, sería: primero la función de escuch de
   body,
135.                luego la función de escucha de div, y por último
   la función
136.                de escucha de button.
```

```
137.
138.                Una vez visto esto, podemos comprender el tercer
   parámetro de addEventListener, que lo que hace es permitirnos
   escoger el orden de propagación:
139.
140.            true: El orden de propagación para el ejemplo sería,
   por tanto,
141.            body-div-button
142.
143.            false: La propagación seguiría el modelo burbuja.
144.            Así, el orden sería button-div-body.
145.
146.            NOTA: omo en nuestro ejemplo utilizamos "false",
   estamos
147.                  eaccionando primero ante el evento sobre las
   teclas,
148.                   posteriormente sobre los eventos asociados al
   CANVAS.
149.                  ste es el mecanismo más usual, pero se utilizará
   "true"
150.                  n las situaciones que lo requieran
151.         */
152.         document.addEventListener("keydown",
   manejadorTeclaPresionada, false);
153.         document.addEventListener("keyup",
   manejadorTeclaLiberada, false);
154.
155.         // Función que maneja tecla presionada
156.         function manejadorTeclaPresionada(e) {
157.             if(e.keyCode == 39) {
158.                     /* e: Es el evento que se produce, en este
   caso
159.                                         tecla presionada. La
   propiedad: keyCode permite
160.                                         descubrir de qué tecla se
   trata. Si el código es 39,
161.                                         se ha presionado la flecha
   derecha. En este caso
162.                                         se coloca la variable:
   flechaDerechaPulsada a true
163.                                 */
164.                 flechaDerechaPulsada = true;
165.             }
166.             else if(e.keyCode == 37) {
167.                     /* e: Es el evento que se produce, en este
   caso
168.                                         tecla presionada. La
   propiedad: keyCode permite
169.                                         descubrir de qué tecla se
   trata. Si el código es 37,
170.                                         se ha presionado la flecha
   izquierda. En este caso
```

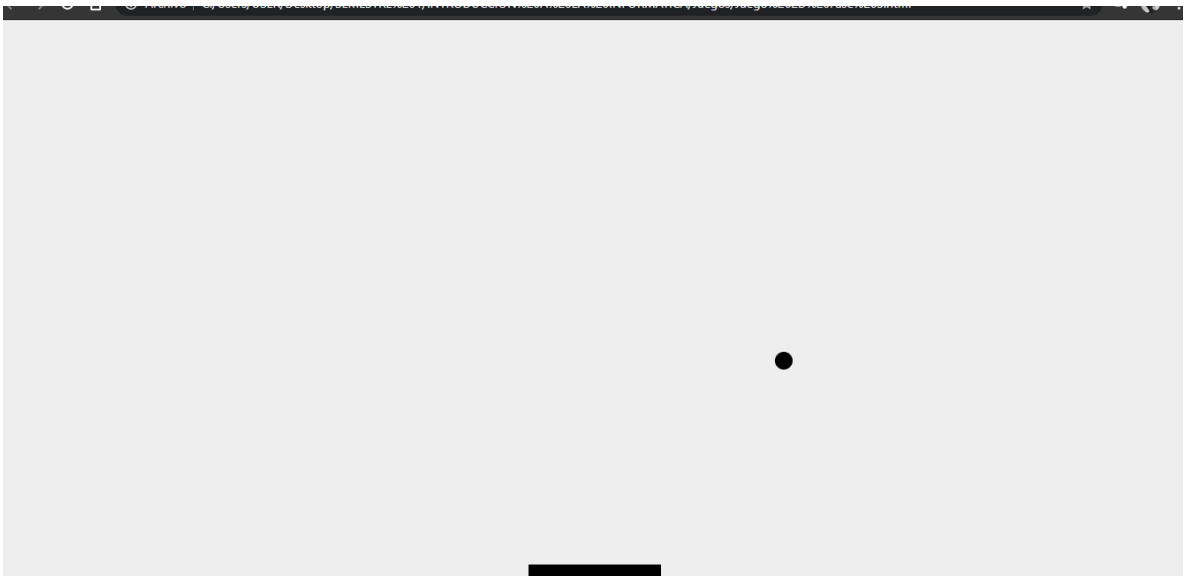
```
171.                                        se coloca la variable:
   flechaIquierdaPulsada a true
172.                                */
173.                flechaIzquierdaPulsada = true;
174.            }
175.        }
176.
177.        // Función que maneja tecla liberada
178.        function manejadorTeclaLiberada(e) {
179.            if(e.keyCode == 39) {
180.                    /* Si la tecla liberada es la 39, se ha
   dejado de
181.                       presionar la flecha derecha. En este caso,
   la variable
182.                       se pone en: false
183.                    */
184.                flechaDerechaPulsada = false;
185.            }
186.            else if(e.keyCode == 37) {
187.                    /* Si la tecla liberada es la 37, se ha
   dejado de
188.                       presionar la flecha izquierda. En este
   caso, la variable
189.                       se pone en: false
190.                    */
191.                flechaIzquierdaPulsada = false;
192.            }
193.        }
194.
195.        // Dibuja la bola. Código explicado en anteriores
   programas
196.        function dibujarBola() {
197.            ctx.beginPath();
198.            ctx.arc(x, y, radioBola, 0, Math.PI*2);
199.            ctx.fillStyle = "#0095DD";
200.            ctx.fill();
201.            ctx.closePath();
202.        }
203.
204.        function dibujarPaleta() {
205.            // Se inicia el dibujo de la paleta
206.            ctx.beginPath();
207.            /* Se crea un rectángulo utilizando la posición en X
208.               El valor de Y está en la base de la pantalla menos
   la
209.               altura de la paleta
210.               Y a continuación se indica la anchura y la altura
   de la paleta
211.                    */
212.            ctx.rect(paletaPosX, canvas.height-alturaPaleta,
   anchuraPaleta, alturaPaleta);
213.            ctx.fillStyle = "#0095DD";
214.            ctx.fill();
```

```
215.             // Se "cierra" la paleta, terminando su dibujo
216.             ctx.closePath();
217.         }
218.
219.         // Función principal. A partir de aquí se origina el
   proceso
220.         // general del juego
221.         function dibujar() {
222.             ctx.clearRect(0, 0, canvas.width, canvas.height);
223.
224.             // En primer lugar, dibuja la bola
225.             dibujarBola();
226.
227.             // Seguidamente, dibuja la paleta
228.             dibujarPaleta();
229.
230.             /* IMPORTANTE:
231.
232.                EL OPERADOR || es el operador lógico OR
233.                Este operador se utiliza para indicar la condición
   de conjunción
234.                SI SE CUMPLE UNA CONDICIÓN, O SE CUMPLE OTRA
   CONDICIÓN, ENTONCES
235.                SE CUMPLE LA CONDICIÓN
236.
237.                EL OPERADOR && es el oerador lógico AND
238.               Este operador se utiliza para indicar la condicion
   de disyunción
239.               SI SE CUMPLE UNA CONDICIÓN, Y SE CUMPLE OTRA
   CONDICIÓN (simultánea), ENTONCES
240.               SE CUMPLE LA CONDICIÓN
241.
242.            */
243.
244.            // Aquí se controla los límites a los que puede llegar
   la bola
245.            // En caso de intentar sobrepasar dichos límites, se
   cambia
246.            // el sentido del movimiento
247.            // Este código se explicó en el anterior programa
248.            if(x + dx > canvas.width-radioBola || x + dx <
   radioBola) {
249.                dx = -dx;
250.             }
251.             if(y + dy > canvas.height-radioBola || y + dy <
   radioBola) {
252.                dy = -dy;
253.            }
254.
255.            /* Si se ha pulsado la flecha derecha, y la paleta aún
   puede
256.               desplazarse hacia la derecha sin que se sobrepase
   el límite de la
```

```
257.                pantalla, entonces se procede a cambiar su posición
258.                En este caso, la función: dibujarPaleta (la cual se
   ejecuta de
259.                manera cíclica) redibujará la paleta en la nueva
   posición
260.             */
261.             if(flechaDerechaPulsada && paletaPosX < canvas.width-
   anchuraPaleta) {
262.                       // Se desplaza la paleta hacia la derecha
263.                       // Aquí, paletaPosX += 7 equivale a:
   paletaPosX = paletaPosX + 7
264.                 paletaPosX += 7;
265.             }
266.             else if(flechaIzquierdaPulsada && paletaPosX > 0) {
267.                       // Se desplaza la paleta hacia la izquierda
268.                       // Aquí, paletaPosX -= 7 equivale a:
   paletaPosX = paletaPosX - 7
269.                  paletaPosX -= 7;
270.              }
271.
272.              x += dx;
273.              y += dy;
274.          }
275.
276.          /* Con esta instrucción se crea un ciclo. Cada 10
   milisegundos se
277.             ejecuta la funcion: dibujar(). Esto genera el ciclo
   que permitirá
278.             actualizar el juego, detectar eventos y cambiar el
   estado
279.             de los objetos según las nuevas posiciones que ocupen
   los
280.             elementos del juego
281.
282.             NOTA: La función que se ejecuta es: dibujar
283.                   Por tanto, dicha función es la encargada de
   "lanzar" el juego
284.                   y dentro de ella se realizarán las acciones que
   desencadenan
285.                   el juego como tal
286.          */
287.          setInterval(dibujar, 10);
288.      </script>
289.
290.      </body>
291.      </html>
```

Gráfica 6. Código de paleta.

Al ejecutar este código se obtiene la siguiente interfaz visual:



Gráfica 7. Ejecución de paleta.

En la figura 7 podemos observar la bola y la paleta en la parte inferior del juego creadas anteriormente en la parte número 3 del código.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

6 FASE 4: FIN DEL JUEGO

En esta parte del programa programaremos que se pueda detectar cuando la bola toca la base de la pantalla, en una coordenada diferente a la de donde se encuentra la paleta, lo que hará que el juego se pierda.

Para este caso analizaremos un código en la función dibujar, el código seria: (y + dy > canvas.height - radioBola) el cual se utilizaría para cuando la bola toque la parte inferior del juego lo cual haría que el juego se pierda. Pero para estar seguros de que el juego se ha perdido analizaremos el siguiente codigo: (x > paletaPosX && x < paletaPosX + anchuraPaleta) el cual hace que se analice la posición de la bola y en caso de que la bola toque la parte inferior hace que se detenga el ciclo de animación del juego y se pierda.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #04 - Game Over</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.      <canvas id="miCanvas" width="480" height="320"></canvas>
11.
12.      <script>
13.              /* Este programa detecta cuando la bola toca la base
   de la pantalla
14.                 Lo anterior significa que la paleta está en otra
   posición distinta
15.                 al punto de toque de la bola con la base de la
   pantalla
16.                 En este caso, se considera que el jugador ha
   perdido una vida
17.                 El sistema lo informa generando una alerta
18.                 El código se encuentra dentro de la función
   dibujar
19.             */
20.
21.          var canvas = document.getElementById("miCanvas");
22.          var ctx = canvas.getContext("2d");
23.
24.          var radioBola = 10;
25.          var x = canvas.width/2;
26.          var y = canvas.height-30;
```

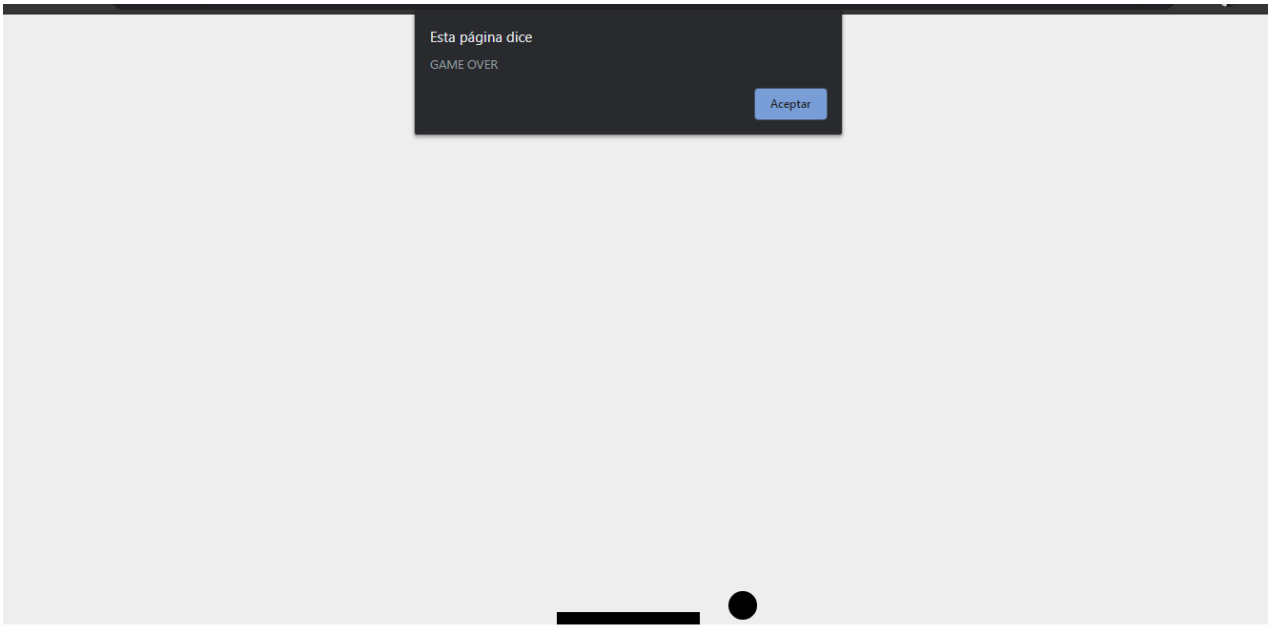
```
27.           var dx = 2;
28.           var dy = -2;
29.
30.           var alturaPaleta = 10;
31.           var anchuraPaleta = 75;
32.           var paletaPosX = (canvas.width-anchuraPaleta)/2;
33.
34.           var flechaDerechaPresionada = false;
35.           var flechaIzquierdaPresionada = false;
36.
37.           document.addEventListener("keydown",
   manejadorTeclaPresionada, false);
38.           document.addEventListener("keyup",
   manejadorTeclaLiberada, false);
39.
40.           function manejadorTeclaPresionada(e) {
41.               if(e.keyCode == 39) {
42.                   flechaDerechaPresionada = true;
43.               }
44.               else if(e.keyCode == 37) {
45.                   flechaIzquierdaPresionada = true;
46.               }
47.           }
48.           function manejadorTeclaLiberada(e) {
49.               if(e.keyCode == 39) {
50.                   flechaDerechaPresionada = false;
51.               }
52.               else if(e.keyCode == 37) {
53.                   flechaIzquierdaPresionada = false;
54.               }
55.           }
56.
57.           function dibujarBola() {
58.               ctx.beginPath();
59.               ctx.arc(x, y, radioBola, 0, Math.PI*2);
60.               ctx.fillStyle = "#0095DD";
61.               ctx.fill();
62.               ctx.closePath();
63.           }
64.           function dibujarPaleta() {
65.               ctx.beginPath();
66.               ctx.rect(paletaPosX, canvas.height-alturaPaleta,
   anchuraPaleta, alturaPaleta);
67.               ctx.fillStyle = "#0095DD";
68.               ctx.fill();
69.               ctx.closePath();
70.           }
71.
72.           function dibujar() {
73.               ctx.clearRect(0, 0, canvas.width, canvas.height);
74.
75.               dibujarBola();
76.               dibujarPaleta();
```

```
77.
78.              if(x + dx > canvas.width-radioBola || x + dx <
   radioBola) {
79.                  dx = -dx;
80.              }
81.              if(y + dy < radioBola) {
82.                  dy = -dy;
83.              }
84.
85.              /* Si y + dy alcanza la frontera inferior de la
   pantalla
86.                 (y + dy > canvas.height - radioBola)
87.                  existe la posibilidad de que el jugador pierda el
   juego
88.                  Para ello debe evaluarse una segunda opción:
89.                  La variable x determina la posición de la bola
90.                  Lo que debe hacerse es mirar si x está DENTRO de
   la palata:
91.                  (x > paletaPosX && x < paletaPosX + anchuraPaleta)
92.                          ----------------------------------------
   -----------------
93.                          Si x está dentro de la paleta, todo va
   bien y se incrementa y
94.                          ----------------------------------------
   -----------------
95.                          Si x NO ESTÁ dentro de la paleta (else),
   la bola ha llegado
96.                          a la frontera inferior, y no encuentra la
   paleta en su camino
97.                          En este caso, SE DETIENE EL CICLO DE
   ANIMACIÓN, y se genera
98.                          un ALERT indicando que el jugador ha
   perdido (GAME OVER)
99.              */
100.             else if(y + dy > canvas.height-radioBola) {
101.                 if(x > paletaPosX && x < paletaPosX +
   anchuraPaleta) {
102.                     dy = -dy;
103.                 }
104.                 else {
105.                     clearInterval(juego);
106.                     alert("GAME OVER");
107.                     document.location.reload();
108.                 }
109.             }
110.
111.             if(flechaDerechaPresionada && paletaPosX <
   canvas.width-anchuraPaleta) {
112.                 paletaPosX += 7;
113.             }
114.             else if(flechaIzquierdaPresionada && paletaPosX > 0)
   {
115.                 paletaPosX -= 7;
```

```
116.             }
117.
118.             x += dx;
119.             y += dy;
120.         }
121.
122.         /* En este programa se asigna a una variable el proceso
   cíclico
123.            Esto tiene mucha importancia, porque si en algún
   momento se requiere
124.            eliminar el ciclo, se utilizará la variable asignada
125.         */
126.         var juego = setInterval(dibujar, 10);
127.     </script>
128.
129.     </body>
130.     </html>
```

Gráfica 8. Código para indicar que se ha perdido el juego.

Al ejecutar este código se obtiene la siguiente interfaz visual:



Gráfica 9. Mostrar fin del juego.

En la figura 9 podemos observar como la bola al tocar la parte inferior del juego y al estar en una coordenada diferente a la paleta aparece un “Game Over” que significa que el juego se ha perdido y se ha acabado, pero si desea repetir el juego solo debe presionar la tecla “aceptar” y el juego se reiniciara inmediatamente.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

7 FASE 5: MURO DE LADRILLOS

En esta parte del juego crearemos unas variables las cuales crearan un muro de ladrillos dentro del juego en los cuales rebotara la bola.

Analizaremos la siguiente función: function dibujarLadrillos() , esta función se apoya de varias variables para la creación del muro de los ladrillos la cual la hace analizando la columna y la fila en la que quedara asignado cada ladrillo.

El siguiente código se muestran las variables de ladrillos:

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D: #05 - Construcción de los ladrillos</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.      <canvas id="miCanvas" width="480" height="320"></canvas>
11.
12.      <script>
13.          var canvas = document.getElementById("miCanvas");
14.          var ctx = canvas.getContext("2d");
15.          var radioBola = 10;
16.          var x = canvas.width/2;
17.          var y = canvas.height-30;
18.          var dx = 2;
19.          var dy = -2;
20.          var alturaPaleta = 10;
21.          var anchuraPaleta = 75;
22.          var paletaPosX = (canvas.width-anchuraPaleta)/2;
23.          var flechaDerechaPresionada = false;
24.          var flechaIzquierdaPresionada = false;
25.
26.          /* NUEVAS VARIABLES asociadas a los ladrillos
27.          */
28.          var nroFilasLadrillos = 5;
29.          var nroColumnasLadrillos = 3;
30.          var anchoLadrillo = 75;
31.          var alturaLadrillo = 20;
32.          var rellenoLadrillo = 10;
33.          var vacioSuperiorLadrillo = 30;
34.          var vacioIzquierdoLadrillo = 30;
35.
```

```
36.          // Crea el conjunto de ladrillos. Inicialmente, vacío
37.          var ladrillos = [];
38.
39.          // Recorre cinco columnas
40.          for(var columna=0; columna<nroColumnasLadrillos;
   columna++) {
41.              // Define la primera columna. Es una lista vertical
42.              ladrillos[columna] = [];
43.
44.              // Para la columna, recorre las tres filas, una
   después de otra
45.              for(var fila=0; fila<nroFilasLadrillos; fila++) {
46.                  // Para cada (columna, fila) se define un ladrillo
47.
48.
49.                  /* IMPORTANTE:
50.                     Como se puede observar, cada ladrillo está
   definido como: ==> ladrillos[c][f]
51.                     Los valores c y f, se corresponden con la fila
   y la columna, DENTRO
52.                     DE LA MATRIZ DE LADRILLOS
53.                     ----------------------------------------------
   ------------------------------
54.                     A cada ladrillo en la posicion (c, f), se le
   asignan tres valores:
55.
56.                        x: Su coordenada horizontal EN LA PANTALLA
57.                        y: Su coordenada vertical EN LA PANTALLA
58.
59.                     ----------------------------------------------
   ------------------------------
60.                     Los valores x y y valen originalmente cero (0)
61.                     Esto cambia cuando se dibujan (más adelante,
   en la función: dibujarLadrillos())
62.                  */
63.                  ladrillos[columna][fila] = { x: 0, y: 0 };
64.              }
65.          }
66.
67.          document.addEventListener("keydown",
   manejadorTeclaPresionada, false);
68.          document.addEventListener("keyup",
   manejadorTeclaLiberada, false);
69.
70.          function manejadorTeclaPresionada(e) {
71.              if(e.keyCode == 39) {
72.                  flechaDerechaPresionada = true;
73.              }
74.              else if(e.keyCode == 37) {
75.                  flechaIzquierdaPresionada = true;
76.              }
77.          }
78.          function manejadorTeclaLiberada(e) {
```

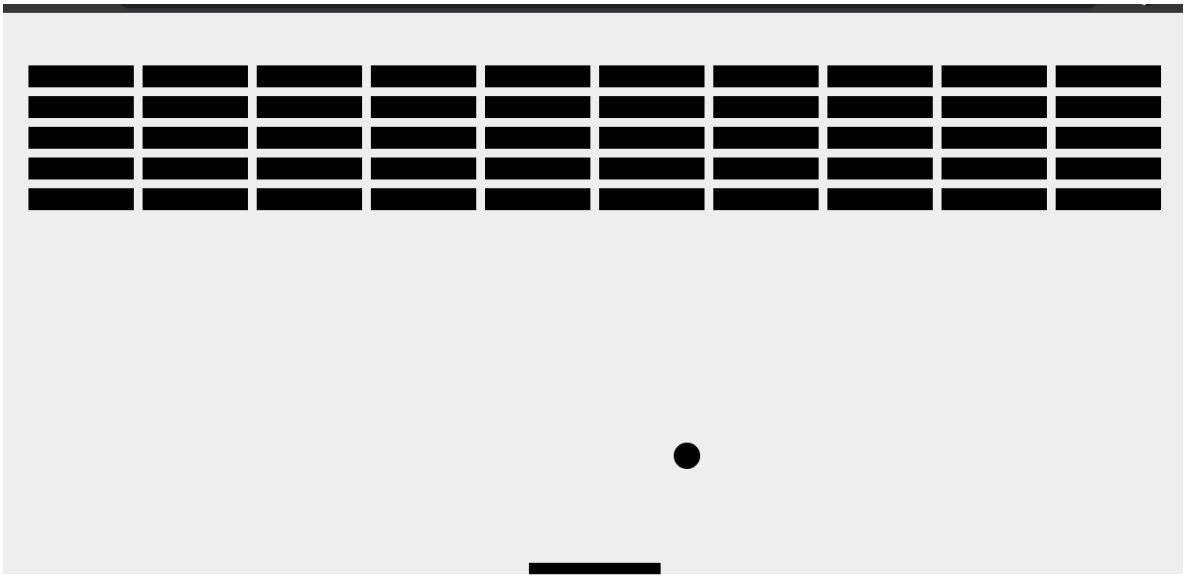
```
79.             if(e.keyCode == 39) {
80.                 flechaDerechaPresionada = false;
81.             }
82.             else if(e.keyCode == 37) {
83.                 flechaIzquierdaPresionada = false;
84.             }
85.         }
86.
87.         function dibujarBola() {
88.             ctx.beginPath();
89.             ctx.arc(x, y, radioBola, 0, Math.PI*2);
90.             ctx.fillStyle = "#0095DD";
91.             ctx.fill();
92.             ctx.closePath();
93.         }
94.
95.         function dibujarPaleta() {
96.             ctx.beginPath();
97.             ctx.rect(paletaPosX, canvas.height-alturaPaleta,
   anchuraPaleta, alturaPaleta);
98.             ctx.fillStyle = "#0095DD";
99.             ctx.fill();
100.            ctx.closePath();
101.        }
102.
103.        /* FUNCIÓN QUE DIBUJA LOS LADRILLOS
104.           -------------------------------
105.        */
106.        function dibujarLadrillos() {
107.            // Recorre todas las columnas
108.            for(var columna=0; columna<nroColumnasLadrillos;
   columna++) {
109.                // Para cada columna, recorre sus filas
110.                for(var fila=0; fila<nroFilasLadrillos; fila++) {
111.                    // Calcula la coordenada x del ladrillo, según
   en que fila se encuentre
112.                    // según el ancho del ladrillo, el valor de
   relleno interno
113.                    // y el espacio que debe dejar a la izquierda
114.                    // NOTA: Se sugiere asignar valores y dibujar
   el esquema a mano
115.                    var
   brickX = (fila*(anchoLadrillo+rellenoLadrillo))+vacioIzquierdoLadri
   llo;
116.
117.                    // Repite el proceso para calcular la
   coordenada y del ladrillo
118.                    var
   brickY = (columna*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLa
   drillo;
119.
120.                    // ASIGNA AL LADRILLO EN LA columna, fila QUE
   LE CORRESPONDE EN LA MATRIZ
```

```
121.                     // EL VALOR CALCULADO (brickX) A SU COORDENADA
   x
122.                     ladrillos[columna][fila].x = brickX;
123.
124.                     // IGUAL PARA EL VALOR y EN PANTALLA
125.                     ladrillos[columna][fila].y = brickY;
126.
127.                     // DIBUJA EL LADRILLO CON LOS VALORES
   ASOCIADOS:
128.                     // Coordenada: (brickX, brickY)
129.                     // Anchura: anchoLadrillo
130.                     // Altrua: alturaLadrillo
131.                     ctx.beginPath();
132.                     ctx.rect(brickX, brickY, anchoLadrillo,
   alturaLadrillo);
133.                     ctx.fillStyle = "#0095DD";
134.                     ctx.fill();
135.                     ctx.closePath();
136.
137.                     // COMO SE RECORRE TODO EL CICLO, SE DIBUJAN
   TODOS LOS LADRILLOS
138.                 }
139.             }
140.         }
141.
142.         function dibujar() {
143.             ctx.clearRect(0, 0, canvas.width, canvas.height);
144.
145.             // DIBUJA EL CONJUNTO DE LADRILLOS
146.             dibujarLadrillos();
147.
148.             dibujarBola();
149.             dibujarPaleta();
150.
151.             if(x + dx > canvas.width-radioBola || x + dx <
   radioBola) {
152.                 dx = -dx;
153.             }
154.             if(y + dy < radioBola) {
155.                 dy = -dy;
156.             }
157.             else if(y + dy > canvas.height-radioBola) {
158.                 if(x > paletaPosX && x < paletaPosX +
   anchuraPaleta)  {
159.                     dy = -dy;
160.                 }
161.                 else {
162.                     clearInterval(juego);
163.                     alert("GAME OVER");
164.
165.                     // RECARGA LA PÁGINA - El juego vuelve a
   empezar
166.                     document.location.reload();
```

```
167.                 }
168.             }
169.
170.             if(flechaDerechaPresionada && paletaPosX <
   canvas.width-anchuraPaleta) {
171.                paletaPosX += 7;
172.             }
173.             else if(flechaIzquierdaPresionada && paletaPosX > 0)
   {
174.                paletaPosX -= 7;
175.             }
176.
177.             x += dx;
178.             y += dy;
179.         }
180.
181.         var juego = setInterval(dibujar, 10);
182.     </script>
183.
184.     </body>
185.     </html>
```

Gráfica 10. Creación de ladrillos

Al ejecutar este código se obtiene la siguiente interfaz visual:



Gráfica 11. Creación ladrillos.

En la figura 11 podemos observar la creación de la pared de ladrillos dentro del campo del juego, tanto columnas como filas. En esta fase la bola aun no puede destruir los ladrillos.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

8 FASE 6: DETECCIÓN DE COLISIONES

En esta parte del programa realizaremos la función que hará que se detecte la colisión de la bola con alguno de los ladrillos y al ocurrir esto hará que el ladrillo con el que colisiono desaparezca.

Procederemos a analizar la función que hace esto posible, la función será la siguiente: function deteccionColision(), esta es la función que permite que cuando la bola colisione con alguno de los ladrillos desaparezca, esto se realiza creando una variable temporal en la cual se asigna el ladrillo y analizando su columna y su fila y asi saber si fue impactado.

```
1. Tambien se crean las siguientes variables: la primera es clearInterval(juego); la cual hace
   que se detenga el ciclo del juego, otra es alert("GAME OVER"); la cual hace que al perder
   el juego salga un letrero con la palabra GAME OVER que significa que se ha acabado el
   juego y por ultimo la siguiente variable document.location.reload(); que hace que el juego
   se recargue nuevamente y se pueda volver a empezar <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #06 - Detección de colisión</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.      <canvas id="miCanvas" width="480" height="320"></canvas>
11.
12.      <script>
13.          var canvas = document.getElementById("miCanvas");
14.          var ctx = canvas.getContext("2d");
15.
16.          var radioBola = 10;
17.          var x = canvas.width/2;
18.          var y = canvas.height-30;
19.          var dx = 2;
20.          var dy = -2;
21.
22.          var alturaPaleta = 10;
23.          var anchuraPaleta = 75;
24.          var paletaPosX = (canvas.width-anchuraPaleta)/2;
25.
26.          var flechaDerechaPresionada = false;
27.          var flechaIzquierdaPresionada = false;
28.
29.          var nroFilasLadrillos = 5;
```

```
30.          var nroColumnasLadrillos = 3;
31.          var anchuraLadrillo = 75;
32.          var alturaLadrillo = 20;
33.          var rellenoLadrillo = 10;
34.          var vacioSuperiorLadrillo = 30;
35.          var vacioIzquierdoLadrillo = 30;
36.
37.          var ladrillos = [];
38.          for(var c=0; c<nroColumnasLadrillos; c++) {
39.             ladrillos[c] = [];
40.             for(var f=0; f<nroFilasLadrillos; f++) {
41.
42.                      /* IMPORTANTE:
43.                                  Como se puede observar, cada
   ladrillo está definido como: ==> ladrillos[c][f]
44.                                  Los valores c y f, se corresponden
   con la fila y la columna, DENTRO
45.                                  DE LA MATRIZ DE LADRILLOS
46.                                  ----------------------------------
   ----------------------------------------
47.                                  A cada ladrillo en la posicion (c,
   f), se le asignan tres valores:
48.
49.                                     x: Su coordenada horizontal EN
   LA PANTALLA
50.                                     y: Su coordenada vertical EN LA
   PANTALLA
51.                                     status: Indica si está visible
   o invisible. 1 = Visible, 0 = INVISIBLE
52.
53.                                  Inicialmente el ladrillo debe esta
   visible. Si la bola "toca" al ladrillo,
54.                                  el ladrillo se debe volver
   INVISIBLE (status = 0)
55.                                  ----------------------------------
   ----------------------------------------
56.                                  Los valores x y y valen
   originalmente cero (0)
57.                                  Esto cambia cuando se dibujan (más
   adelante, en la función: dibujarLadrillos())
58.                               */
59.                  ladrillos[c][f] = { x: 0, y: 0, status: 1 };
60.              }
61.          }
62.
63.          document.addEventListener("keydown",
   manejadorTeclaPresionada, false);
64.          document.addEventListener("keyup",
   manejadorTeclaLiberada, false);
65.
66.          function manejadorTeclaPresionada(e) {
67.              if(e.keyCode == 39) {
68.                  flechaDerechaPresionada = true;
```

```
69.              }
70.              else if(e.keyCode == 37) {
71.                  flechaIzquierdaPresionada = true;
72.              }
73.          }
74.
75.          function manejadorTeclaLiberada(e) {
76.              if(e.keyCode == 39) {
77.                  flechaDerechaPresionada = false;
78.              }
79.              else if(e.keyCode == 37) {
80.                  flechaIzquierdaPresionada = false;
81.              }
82.          }
83.
84.          // EN ESTA FUNCIÓN SE DETECTA LA COLISIÓN DE LA BOLA CON
   EL LADRILLO
85.
86.          function deteccionColision() {
87.
88.              // LOS DOS CICLOS SIGUIENTES RECORREN TODOS LOS
   LADRILLOS
89.              for(var c=0; c<nroColumnasLadrillos; c++) {
90.                 for(var f=0; f<nroFilasLadrillos; f++) {
91.
92.                      // EN ESTE PUNTO SE TIENE EL LADRILLO SITUADO
   EN: (c, f)
93.                      // SE CREA UNA VARIABLE TEMPORAL PARA EL
   LADRILLO
94.                     var b = ladrillos[c][f];
95.
96.                     // SI EL LADRILLO ES VISIBLE, se debe
   verificar si entra en contacto con la bola
97.                     if(b.status == 1) {
98.
99.                                 /* SI LAS COORDENADAS x y y, SE
   ENCUENTRAN DENTRO DE LAS COORDENADAS
100.                                DEL LADRILLO (aspecto que se
   verifica con las condiciones mostradas)
101.                                LA BOLA HA IMPACTADO CONTRA EL
   LADRILLO
102.                                En este caso, se modifica la
   coordenada y, PERO LÓ MÁS IMPORTANTE
103.                                ES QUE SE COLOCA EL VALOR DE
   status A CERO, HACIENDO QUE EL LADRILLO
104.                                SE VUELVA INVISIBLE
105.                             */
106.
107.                        if(x > b.x && x < b.x+anchuraLadrillo && y
   > b.y && y < b.y+alturaLadrillo) {
108.                            dy = -dy;
109.                             b.status = 0;
110.                         }
```

```
111.                     }
112.                 }
113.             }
114.         }
115.
116.         function dibujarBola() {
117.             ctx.beginPath();
118.             ctx.arc(x, y, radioBola, 0, Math.PI*2);
119.             ctx.fillStyle = "#0095DD";
120.             ctx.fill();
121.             ctx.closePath();
122.         }
123.
124.         function dibujarPaleta() {
125.             ctx.beginPath();
126.             ctx.rect(paletaPosX, canvas.height-alturaPaleta,
    anchuraPaleta, alturaPaleta);
127.             ctx.fillStyle = "#0095DD";
128.             ctx.fill();
129.             ctx.closePath();
130.         }
131.
132.         function dibujarLadrillos() {
133.             for(var c=0; c<nroColumnasLadrillos; c++) {
134.                 for(var f=0; f<nroFilasLadrillos; f++) {
135.
136.                     /* IMPORTANTE:
137.
138.                        Solamente se dibujan los ladrillos que
    están VISIBLES
139.                        Se sabe que el ladrillo es visible cuando:
    status == 1
140.                        Los ladrillos INVISIBLES NO SE DIBUJAN
141.
142.                     */
143.
144.                     if(ladrillos[c][f].status == 1) {
145.
146.                             // SE DIBUJA EL LADRILLO
147.                         var
    brickX = (f*(anchuraLadrillo+rellenoLadrillo))+vacioIzquierdoLadril
    lo;
148.                         var
    brickY = (c*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLadrillo
    ;
149.                         ladrillos[c][f].x = brickX;
150.                         ladrillos[c][f].y = brickY;
151.                         ctx.beginPath();
152.                         ctx.rect(brickX, brickY, anchuraLadrillo,
    alturaLadrillo);
153.                         ctx.fillStyle = "#0095DD";
154.                         ctx.fill();
155.                         ctx.closePath();
```

```
156.                     }
157.                 }
158.             }
159.         }
160.
161.         function dibujar() {
162.             ctx.clearRect(0, 0, canvas.width, canvas.height);
163.             dibujarLadrillos();
164.             dibujarBola();
165.             dibujarPaleta();
166.             deteccionColision();
167.
168.             if(x + dx > canvas.width-radioBola || x + dx <
    radioBola) {
169.                 dx = -dx;
170.             }
171.             if(y + dy < radioBola) {
172.                 dy = -dy;
173.             }
174.             else if(y + dy > canvas.height-radioBola) {
175.                 if(x > paletaPosX && x < paletaPosX +
    anchuraPaleta) {
176.                     dy = -dy;
177.                 }
178.                 else {
179.                     // Detiene el ciclo del juego
180.                     clearInterval(juego);
181.                     // Genera mensaje, pues el jugador ha perdido
182.                     alert("GAME OVER");
183.                     // Recarga la página, para iniciar de nuevo
    el juego
184.                     document.location.reload();
185.                 }
186.             }
187.
188.             if(flechaDerechaPresionada && paletaPosX <
    canvas.width-anchuraPaleta) {
189.                 paletaPosX += 7;
190.             }
191.             else if(flechaIzquierdaPresionada && paletaPosX > 0)
    {
192.                 paletaPosX -= 7;
193.             }
194.
195.             x += dx;
196.             y += dy;
197.         }
198.
199.         var juego = setInterval(dibujar, 10);
200.     </script>
201.
202.     </body>
203.     </html>
```

Al ejecutar este código se obtiene la siguiente interfaz visual:

Gráfica 12. Destrucción de ladrillos.

En la figura 12 podemos observar como algunos ladrillos se desaparecieron luego de ser golpeados por la bola.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

9 FASE 7: CONTAR PUNTOS Y GANAR

En esta parte del programa se realiza la variable para darle algún valor cuando la bola golpee algún ladrillo y se convierta en un punto y se sumen estos puntos hasta ganar el juego con el máximo de puntos que se puedan obtener

Se crea una variable llamada puntaje la cual controla la cantidad de ladrillos que han sido golpeados por la bola, cada que la bola impacta un ladrillo se le agrega un valor a esta variable hasta que el puntaje es igual al número de ladrillos haciendo que el juego se gane.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #07 - Control de juego ganado</title>
6.     <!-- EN ESTE EJEMPLO SE CAMBIA LA ANCHURA DE LA PALETA
7.          ESTO ES CLAVE PARA PERMITIR QUE EL JUEGO SEA AUTOMÁTICO
8.          Y SE PUEDA VERIFICAR EL OBJETIVO DEL JUEGO Y EL JUGADOR
   GANE -->
9.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
10.      </head>
11.      <body>
12.
13.      <canvas id="miCanvas" width="480" height="320"></canvas>
14.
15.      <script>
16.          var canvas = document.getElementById("miCanvas");
17.          var ctx = canvas.getContext("2d");
18.
19.          var radioBola = 10;
20.          var x = canvas.width/2;
21.          var y = canvas.height-30;
22.          var dx = 2;
23.          var dy = -2;
24.          var alturaPaleta = 10;
25.
26.          // EL ANCHO DE LA PALETA ES 480. ESTE ES EL MISMO ANCHO
   DEL CANVAS
27.          // Con esto se garantiza que el juego termine
28.          var anchuraPaleta = 480;
29.
30.          var paletaPosX = (canvas.width-anchuraPaleta)/2;
31.          var flechaDerechaPresionada = false;
32.          var flechaIzquierdaPresionada = false;
33.
34.          var nroFilasLadrillos = 5;
```

```
35.          var nroColumnasLadrillos = 3;
36.          var anchuraLadrillos = 75;
37.          var alturaLadrillos = 20;
38.          var rellenoLadrillos = 10;
39.          var vacioSuperiorLadrillo = 30;
40.          var vacioIzquierdoLadrillo = 30;
41.
42.          // LA VARIABLE puntaje CONTROLA EL NÚMERO DE LADRILLOS
   QUE HAN SIDO
43.          // IMPACTADOS POR LA BOLA. Cada vez que la bola golpee un
   ladrillo,
44.          // la variable "puntaje" se incrementa en uno
45.          var puntaje = 0;
46.
47.          var ladrillos = [];
48.          for(var c=0; c<nroColumnasLadrillos; c++) {
49.             ladrillos[c] = [];
50.             for(var f=0; f<nroFilasLadrillos; f++) {
51.                 ladrillos[c][f] = { x: 0, y: 0, estado: 1 };
52.             }
53.         }
54.
55.         document.addEventListener("keydown",
   manejadorTeclaPresionada, false);
56.         document.addEventListener("keyup", manejadorTeclaLiberada,
   false);
57.
58.         function manejadorTeclaPresionada(e) {
59.             if(e.keyCode == 39) {
60.                 flechaDerechaPresionada = true;
61.             }
62.             else if(e.keyCode == 37) {
63.                 flechaIzquierdaPresionada = true;
64.             }
65.         }
66.         function manejadorTeclaLiberada(e) {
67.             if(e.keyCode == 39) {
68.                 flechaDerechaPresionada = false;
69.             }
70.             else if(e.keyCode == 37) {
71.                 flechaIzquierdaPresionada = false;
72.             }
73.         }
74.         function detectarColision() {
75.             for(var c=0; c<nroColumnasLadrillos; c++) {
76.                 for(var f=0; f<nroFilasLadrillos; f++) {
77.                     var b = ladrillos[c][f];
78.                     if(b.estado == 1) {
79.                         if(x > b.x && x < b.x+anchuraLadrillos
80.      && y > b.y && y < b.y+alturaLadrillos) {
81.                             dy = -dy;
82.                              b.estado = 0;
83.
```

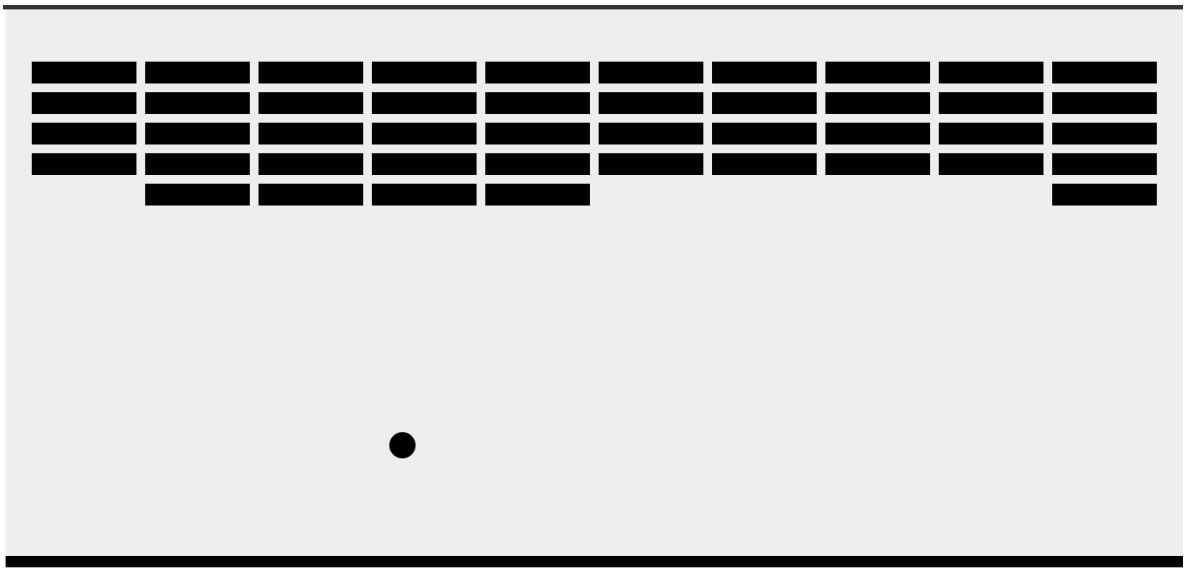
```
84.                                 // LA INSTRUCCIÓN puntaje++ EQUIVALE
   A: puntaje = puntaje + 1
85.                                 // ----------------------------------
   ------------------------
86.                                 // EN ESTE PUNTO DEL CÓDIGO LA BOLA
   HA IMPACTADO UN LADRILLO
87.                                 // POR ESTE MOTIVO, SE INCREMENTA EL
   VALOR DE puntaje
88.                                 // Si el puntaje es igual al número
   total de ladrillos (valor que
89.                                 // se obtiene multiplicando el número
   de filas de ladrillos por el
90.                                 // número de columnas de ladrillos),
   entonces el jugador ha ganado
91.                                 puntaje++;
92.                                 if(puntaje ==
   nroFilasLadrillos*nroColumnasLadrillos) {
93.                                     alert("USTED GANA!
   FELICITACIONES!!!");
94.                                     document.location.reload();
95.                                 }
96.
97.                             }
98.                         }
99.                     }
100.                }
101.            }
102.
103.            function dibujarBola() {
104.                ctx.beginPath();
105.                ctx.arc(x, y, radioBola, 0, Math.PI*2);
106.                ctx.fillStyle = "#0095DD";
107.                ctx.fill();
108.                ctx.closePath();
109.            }
110.
111.            function dibujarPaleta() {
112.                ctx.beginPath();
113.                ctx.rect(paletaPosX, canvas.height-alturaPaleta,
   anchuraPaleta, alturaPaleta);
114.                ctx.fillStyle = "#0095DD";
115.                ctx.fill();
116.                ctx.closePath();
117.            }
118.
119.            function dibujarLadrillos() {
120.                for(var c=0; c<nroColumnasLadrillos; c++) {
121.                   for(var r=0; r<nroFilasLadrillos; r++) {
122.                       if(ladrillos[c][r].estado == 1) {
123.                           var
   posXLadrillo = (r*(anchuraLadrillos+rellenoLadrillos))+vacioIzquier
   doLadrillo;
```

```
124.                         var
    posYLadrillo = (c*(alturaLadrillos+rellenoLadrillos))+vacioSuperior
    Ladrillo;
125.                         ladrillos[c][r].x = posXLadrillo;
126.                         ladrillos[c][r].y = posYLadrillo;
127.                         ctx.beginPath();
128.                         ctx.rect(posXLadrillo, posYLadrillo,
    anchuraLadrillos, alturaLadrillos);
129.                         ctx.fillStyle = "#0095DD";
130.                         ctx.fill();
131.                         ctx.closePath();
132.                     }
133.                 }
134.             }
135.         }
136.
137.         function dibujarPuntaje() {
138.             ctx.font = "16px Arial";
139.             ctx.fillStyle = "#0095DD";
140.             ctx.fillText("puntaje: "+puntaje, 8, 20);
141.         }
142.
143.         function dibujar() {
144.             ctx.clearRect(0, 0, canvas.width, canvas.height);
145.             dibujarLadrillos();
146.             dibujarBola();
147.             dibujarPaleta();
148.             dibujarPuntaje();
149.             detectarColision();
150.
151.             if(x + dx > canvas.width-radioBola || x + dx <
    radioBola) {
152.                 dx = -dx;
153.             }
154.             if(y + dy < radioBola) {
155.                 dy = -dy;
156.             }
157.             else if(y + dy > canvas.height-radioBola) {
158.                 if(x > paletaPosX && x < paletaPosX +
    anchuraPaleta) {
159.                     dy = -dy;
160.                 }
161.                 else {
162.                     clearInterval(juego);
163.                     alert("GAME OVER");
164.                     document.location.reload();
165.                 }
166.             }
167.
168.             if(flechaDerechaPresionada && paletaPosX <
    canvas.width-anchuraPaleta) {
169.                 paletaPosX += 7;
170.             }
```

```
171.             else if(flechaIzquierdaPresionada && paletaPosX > 0)
   {
172.                paletaPosX -= 7;
173.             }
174.
175.             x += dx;
176.             y += dy;
177.         }
178.
179.         var juego = setInterval(dibujar, 10);
180.     </script>
181.
182.     </body>
183.     </html>
```

Gráfica 13. Código, detección de colisión.

Al ejecutar este código se obtiene la siguiente interfaz visual:



Gráfica 14. Creación paleta.

En la figura 14 se puede observar como la bola al impactar en los ladrillos estos desaparecen y el puntaje incrementa hasta desaparecer todos los ladrillos y ganar el juego.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

10 FASE 8: CONTROLANDO EL RATÓN

En esta parte del programa haremos que la paleta en lugar de ser movida por las flechas sea movida por el mouse.

Esto se obtiene creando una función llamada function manejadorRaton(e) a la cual se le da una variable y una condición que al cumplirla hace que la paleta pueda ser desplazada mediante el mouse.

Esta configuración se implementa para que aquellos jugadores que no les gusta mover la paleta con las teclas lo puedan hacer directamente del mouse.

A continuación, veremos el código con sus respectivas modificaciones para la variable del manejador del ratón.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #08 - Utilizando el ratón</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.      <canvas id="miCanvas" width="480" height="320"></canvas>
11.
12.      <script>
13.          var canvas = document.getElementById("miCanvas");
14.          var ctx = canvas.getContext("2d");
15.
16.          var radioBola = 10;
17.          var x = canvas.width/2;
18.          var y = canvas.height-30;
19.          var dx = 2;
20.          var dy = -2;
21.
22.          var alturaPaleta = 10;
23.          var anchuraPaleta = 75;
24.          var paletaPosX = (canvas.width-anchuraPaleta)/2;
25.
26.          var flechaDerechaPresionada = false;
27.          var flechaIzquierdaPresionada = false;
28.
29.          var nroFilasLadrillos = 5;
30.          var nroColumnasLadrillos = 3;
31.          var anchuraLadrillo = 75;
```

```
32.          var alturaLadrillo = 20;
33.          var rellenoLadrillo = 10;
34.          var vacioSuperiorLadrillo = 30;
35.          var vacioIzquierdoLadrillo = 30;
36.
37.          var puntaje = 0;
38.
39.          var ladrillos = [];
40.          for(var c=0; c<nroColumnasLadrillos; c++) {
41.             ladrillos[c] = [];
42.             for(var f=0; f<nroFilasLadrillos; f++) {
43.                 ladrillos[c][f] = { x: 0, y: 0, estado: 1 };
44.             }
45.         }
46.
47.         document.addEventListener("keydown",
    manejadorTeclaPresionada, false);
48.         document.addEventListener("keyup", manejadorTeclaLiberada,
    false);
49.
50.         // PARA DETECTAR EL MOVIMIENTO DEL RATÓN, SE COLOCA UN
    ESCUCHADOR (listener)
51.         // AL EVENTO "mousemove"
52.         document.addEventListener("mousemove", manejadorRaton,
    false);
53.
54.         function manejadorTeclaPresionada(e) {
55.             if(e.keyCode == 39) {
56.                 flechaDerechaPresionada = true;
57.             }
58.             else if(e.keyCode == 37) {
59.                 flechaIzquierdaPresionada = true;
60.             }
61.         }
62.
63.         function manejadorTeclaLiberada(e) {
64.             if(e.keyCode == 39) {
65.                 flechaDerechaPresionada = false;
66.             }
67.             else if(e.keyCode == 37) {
68.                 flechaIzquierdaPresionada = false;
69.             }
70.         }
71.
72.         // ESTE ES EL MANEJADOR DEL RATÓN
73.         // ------------------------------
74.         // La instrucción: "offsetLeft" calcula la distancia desde
    el borde izquierdo
75.         // de la pantalla hasta un componente html
76.         // ------------------------------
77.         // Por tanto, la instrucción: "canvas.offsetLeft" calcula
    el espacio a la izquierda
78.         // del objeto CANVAS
```

```
79.         // ------------------------------
80.         // Dentro del manejador del ratón, la
   instrucción: "e.clientX" calcula la posición
81.         // del ratón en la pantalla. Para calcular la posición del
   ratón DENTRO del CANVAS
82.         // debemos RESTAR a la posición X del ratón, el valor
   izquierdo del CANVAS
83.         // ------------------------------
84.         // Es decir: "e.clientX - canvas.offsetLeft"
85.         // ------------------------------
86.         function manejadorRaton(e) {
87.             var posXRatonDentroDeCanvas = e.clientX -
   canvas.offsetLeft;
88.             // EL SIGUIENTE if DETERMINA SI LA POSICIÓN X DEL
   RATÓN ESTÁ
89.             // DENTRO DEL CANVAS
90.             if(posXRatonDentroDeCanvas > 0 &&
   posXRatonDentroDeCanvas < canvas.width) {
91.                 // SI LA RESPUESTA ES POSITIVA, EL RATÓN ESTÁ
   DENTRO DEL CANVAS
92.                 // EN ESTE CASO, SE RECALCULA LA POSICIÓN DE LA
   PALETA
93.                 // SU VALOR X ES AHORA LA POSICIÓN X DEL RATÓN
94.                 // ---------------------------------------------
   -------------
95.                 // PERO DEBE RECORDARSE QUE LA PALETA TIENE UN
   ANCHO. ESTA ES LA RAZÓN
96.                 // POR LA CUAL SE DEBE RESTAR A LA POSICIPÓN X DE
   LA PALETA LA MITAD DEL
97.                 // ANCHO DE LA PALETA
98.                 // ---------------------------------------------
   -------------
99.                 // AL HACER ESTO, LA PALETA MODIFICA SU POSICIÓN
   CON BASE EN EL
100.                // MOVIMIENTO DEL RATÓN
101.                paletaPosX = posXRatonDentroDeCanvas -
   anchuraPaleta/2;
102.            }
103.        }
104.        function detectarColision() {
105.            for(var c=0; c<nroColumnasLadrillos; c++) {
106.                for(var r=0; r<nroFilasLadrillos; r++) {
107.                    var b = ladrillos[c][r];
108.                    if(b.estado == 1) {
109.                        if(x > b.x && x < b.x+anchuraLadrillo && y
   > b.y && y < b.y+alturaLadrillo) {
110.                            dy = -dy;
111.                            b.estado = 0;
112.                            puntaje++;
113.                            if(puntaje ==
   nroFilasLadrillos*nroColumnasLadrillos) {
114.                                alert("USTED GANA,
   FELICITACIONES!!!");
```

```
115.                                    document.location.reload();
116.                                }
117.                            }
118.                        }
119.                    }
120.                }
121.            }
122.
123.            function dibujarBola() {
124.                ctx.beginPath();
125.                ctx.arc(x, y, radioBola, 0, Math.PI*2);
126.                ctx.fillStyle = "#0095DD";
127.                ctx.fill();
128.                ctx.closePath();
129.            }
130.            function dibujarPaleta() {
131.                ctx.beginPath();
132.                ctx.rect(paletaPosX, canvas.height-alturaPaleta,
    anchuraPaleta, alturaPaleta);
133.                ctx.fillStyle = "#0095DD";
134.                ctx.fill();
135.                ctx.closePath();
136.            }
137.            function dibujarLadrillos() {
138.                for(var c=0; c<nroColumnasLadrillos; c++) {
139.                   for(var r=0; r<nroFilasLadrillos; r++) {
140.                       if(ladrillos[c][r].estado == 1) {
141.                           var
    brickX = (r*(anchuraLadrillo+rellenoLadrillo))+vacioIzquierdoLadril
    lo;
142.                           var
    brickY = (c*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLadrillo
    ;
143.                           ladrillos[c][r].x = brickX;
144.                           ladrillos[c][r].y = brickY;
145.                           ctx.beginPath();
146.                           ctx.rect(brickX, brickY, anchuraLadrillo,
    alturaLadrillo);
147.                           ctx.fillStyle = "#0095DD";
148.                           ctx.fill();
149.                           ctx.closePath();
150.                       }
151.                   }
152.               }
153.           }
154.           function dibujarPuntaje() {
155.               ctx.font = "16px Arial";
156.               ctx.fillStyle = "#0095DD";
157.               ctx.fillText("puntaje: "+puntaje, 8, 20);
158.           }
159.
160.           function dibujar() {
161.               ctx.clearRect(0, 0, canvas.width, canvas.height);
```

```
162.             dibujarLadrillos();
163.             dibujarBola();
164.             dibujarPaleta();
165.             dibujarPuntaje();
166.             detectarColision();
167.
168.             if(x + dx > canvas.width-radioBola || x + dx <
    radioBola) {
169.                 dx = -dx;
170.             }
171.             if(y + dy < radioBola) {
172.                 dy = -dy;
173.             }
174.             else if(y + dy > canvas.height-radioBola) {
175.                  if(x > paletaPosX && x < paletaPosX +
    anchuraPaleta) {
176.                     dy = -dy;
177.                  }
178.                  else {
179.                      clearInterval(juego);
180.                      alert("GAME OVER");
181.                      document.location.reload();
182.                  }
183.              }
184.
185.              if(flechaDerechaPresionada && paletaPosX <
    canvas.width-anchuraPaleta) {
186.                 paletaPosX += 7;
187.              }
188.              else if(flechaIzquierdaPresionada && paletaPosX > 0)
    {
189.                 paletaPosX -= 7;
190.              }
191.
192.              x += dx;
193.              y += dy;
194.          }
195.
196.          var juego = setInterval(dibujar, 10);
197.      </script>
198.
199.      </body>
200.      </html>
```

Gráfica 15. Código, controlador de mouse.

Al ejecutar este código se obtiene la siguiente interfaz visual:

Gráfica 16. Controlador del mouse.

En la figura 16 se puede observar como la paleta es controlada de derecha a izquierda mediante el uso del mouse. Aparte de una mejor comodidad para el gamer, con el uso del mouse se obtiene una mayor velocidad de reacción en la paleta.

En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

11 FASE 9: FINALIZANDO EL JUEGO

En esta parte del programa ya se agregan los toques finales tales como vidas para el jugador, ocultar el mouse entre otras.

Se crea una variable var vidas = 3 con la instrucción de controlar las vidas que tiene dentro del juego cada participante y se crea otra variable canvas.style.cursor = 'none' para ocultar el mouse dentro del campo del juego, también se crea la instrucción vidas--; la cual lleva la cuenta de las vidas que tiene.

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #09 - Juego completo</title>
6.     <!-- 1. Se oculta el ratón
7.          2. Se agregan vidas al jugador
8.          3. Ya no se utiliza "setInterval" -->
9.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; * {cursor: none;} } </style>
10.      </head>
11.      <body>
12.
13.      <canvas id="miCanvas" width="480" height="320"></canvas>
14.
15.      <script>
16.          var canvas = document.getElementById("miCanvas");
17.          var ctx = canvas.getContext("2d");
18.
19.          var bolaRadio = 10;
20.          var x = canvas.width/2;
21.          var y = canvas.height-30;
22.          var dx = 2;
23.          var dy = -2;
24.
25.          var alturaPaleta = 10;
26.          var anchuraPaleta = 75;
27.          var paletaPosX = (canvas.width-anchuraPaleta)/2;
28.
29.          var flechaDerechaPresionada = false;
30.          var flechaIzquierdaPresionada = false;
31.
32.          var nroFilasLadrillos = 5;
33.          var nroColumnasLadrillos = 3;
34.          var anchuraLadrillo = 75;
35.          var alturaLadrillo = 20;
36.          var rellenoLadrillo = 10;
```

```
37.          var vacioSuperiorLadrillo = 30;
38.          var vacioIzquierdoLadrillo = 30;
39.
40.          var puntaje = 0;
41.
42.          // ESTA INSTRUCCIÓN CONTROLA EL NÚMERO DE VIDAS DEL
   JUGADOR
43.          // CUANDO LA INSTRUCCIÓN vidas DISMINUYE A CERO, EL
   JUGADOR PIERDE,
44.          // PUESTO QUE HA PERDIDO TRES VECES
45.          var vidas = 3;
46.
47.          // ESTA VARIABLE DEFINE UN COLOR
48.          // Se pueden utilizar otros colores para los diferentes
   elementos del juego
49.          var colorFigura = "#ff0000";
50.          var colorBola = "#137B13";
51.          var colorPaleta = "#0000ff";
52.          var colorLadrillo = "#dd2244";
53.          var colorTexto = "#000000";
54.
55.          // ESTA INSTRUCCIÓN OCULTA EL CURSOR DEL RATON (DENTRO
   DEL CANVAS)
56.          canvas.style.cursor = 'none';
57.
58.          var ladrillos = [];
59.          for(var c=0; c<nroColumnasLadrillos; c++) {
60.             ladrillos[c] = [];
61.             for(var f=0; f<nroFilasLadrillos; f++) {
62.                 ladrillos[c][f] = { x: 0, y: 0, estado: 1 };
63.             }
64.         }
65.
66.         document.addEventListener("keydown",
   manejadorTeclaPresionada, false);
67.         document.addEventListener("keyup", manejadorTeclaLiberada,
   false);
68.         document.addEventListener("mousemove", manejadorRaton,
   false);
69.
70.         function manejadorTeclaPresionada(e) {
71.             if(e.keyCode == 39) {
72.                 flechaDerechaPresionada = true;
73.             }
74.             else if(e.keyCode == 37) {
75.                 flechaIzquierdaPresionada = true;
76.             }
77.         }
78.
79.         function manejadorTeclaLiberada(e) {
80.             if(e.keyCode == 39) {
81.                 flechaDerechaPresionada = false;
82.             }
```

```
83.             else if(e.keyCode == 37) {
84.                 flechaIzquierdaPresionada = false;
85.             }
86.         }
87.
88.         function manejadorRaton(e) {
89.             var posXRatonDentroDeCanvas = e.clientX -
    canvas.offsetLeft;
90.             if(posXRatonDentroDeCanvas > 0 &&
    posXRatonDentroDeCanvas < canvas.width) {
91.                 paletaPosX = posXRatonDentroDeCanvas -
    anchuraPaleta/2;
92.             }
93.         }
94.
95.         function detectarColision() {
96.             for(var c=0; c<nroColumnasLadrillos; c++) {
97.                 for(var f=0; f<nroFilasLadrillos; f++) {
98.                     var b = ladrillos[c][f];
99.                     if(b.estado == 1) {
100.                        if(x > b.x && x < b.x+anchuraLadrillo && y
    > b.y && y < b.y+alturaLadrillo) {
101.                            dy = -dy;
102.                            b.estado = 0;
103.                            puntaje++;
104.                            if(puntaje ==
    nroFilasLadrillos*nroColumnasLadrillos) {
105.                                alert("USTED GANA,
    FELICITACIONES!");
106.                                document.location.reload();
107.                            }
108.                        }
109.                    }
110.                }
111.            }
112.        }
113.
114.        function dibujarBola() {
115.            ctx.beginPath();
116.            ctx.arc(x, y, bolaRadio, 0, Math.PI*2);
117.            // SE UTILIZA EL COLOR PREVIAMENTE DEFINIDO
118.            ctx.fillStyle = colorBola;
119.            ctx.fill();
120.            ctx.closePath();
121.        }
122.        function dibujarPaleta() {
123.            ctx.beginPath();
124.            ctx.rect(paletaPosX, canvas.height-alturaPaleta,
    anchuraPaleta, alturaPaleta);
125.            ctx.fillStyle = colorPaleta;
126.            ctx.fill();
127.            ctx.closePath();
128.        }
```

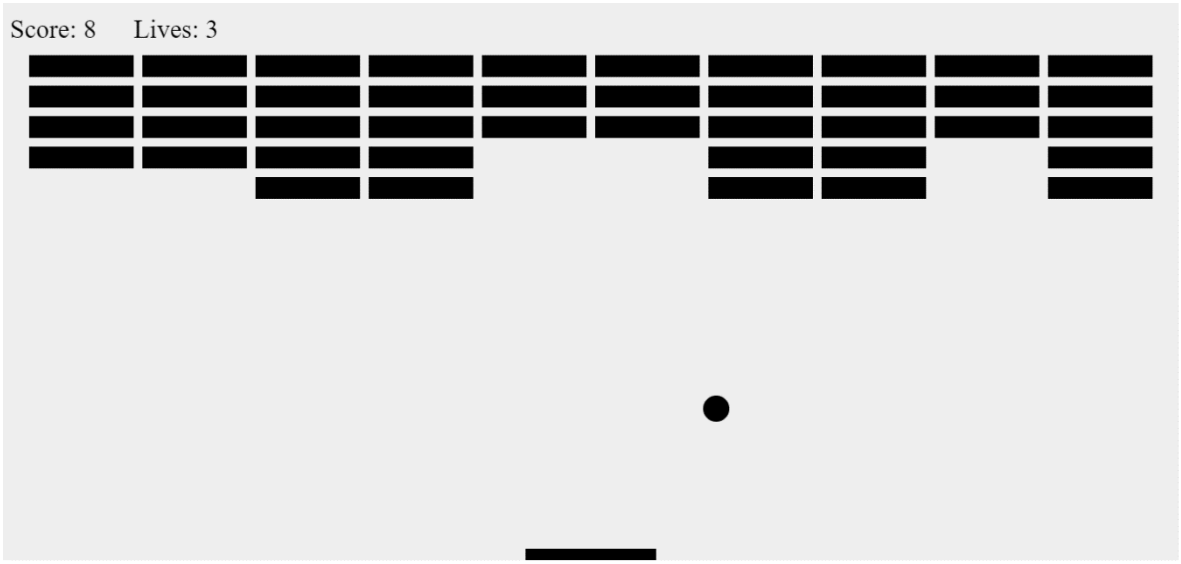
```
129.         function dibujarLadrillos() {
130.             for(var c=0; c<nroColumnasLadrillos; c++) {
131.                 for(var f=0; f<nroFilasLadrillos; f++) {
132.                     if(ladrillos[c][f].estado == 1) {
133.                         var
    ladrilloX = (f*(anchuraLadrillo+rellenoLadrillo))+vacioIzquierdoLad
    rillo;
134.                         var
    ladrilloY = (c*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLadri
    llo;
135.                         ladrillos[c][f].x = ladrilloX;
136.                         ladrillos[c][f].y = ladrilloY;
137.                         ctx.beginPath();
138.                         ctx.rect(ladrilloX, ladrilloY,
    anchuraLadrillo, alturaLadrillo);
139.                         ctx.fillStyle = colorLadrillo;
140.                         ctx.fill();
141.                         ctx.closePath();
142.                     }
143.                 }
144.             }
145.         }
146.
147.         function dibujarPuntaje() {
148.             ctx.font = "16px Arial";
149.             ctx.fillStyle = colorTexto;
150.             ctx.fillText("puntaje: "+puntaje, 8, 20);
151.         }
152.
153.         function dibujarVidas() {
154.             ctx.font = "16px Arial";
155.             ctx.fillStyle = colorTexto;
156.             // SE MUESTRA EL NÚMERO DE VIDAS DISPONIBLES
157.             ctx.fillText("vidas: "+vidas, canvas.width-65, 20);
158.         }
159.
160.         function dibujar() {
161.             ctx.clearRect(0, 0, canvas.width, canvas.height);
162.             dibujarLadrillos();
163.             dibujarBola();
164.             dibujarPaleta();
165.             dibujarPuntaje();
166.             dibujarVidas();
167.             detectarColision();
168.
169.             if(x + dx > canvas.width-bolaRadio || x + dx <
    bolaRadio) {
170.                 dx = -dx;
171.             }
172.             if(y + dy < bolaRadio) {
173.                 dy = -dy;
174.             }
175.             else if(y + dy > canvas.height-bolaRadio) {
```

```
176.                 if(x > paletaPosX && x < paletaPosX +
   anchuraPaleta) {
177.                    dy = -dy;
178.                 }
179.                 else {
180.                     // SI SE PRODUCE UN CONTACTO DE LA BOLA CON
   LA BASE DEL CANVAS
181.                     // SE PIERDE UNA VIDA. PARA ELLO, LA
   INSTRUCCIÓN vidas--;
182.                     // LO CUAL EQUIVALE A: vidas = vidas - 1
183.                     vidas--;
184.                     if(!vidas) {
185.                             // SI vidas == 0 (lo cual también
   puede escribir: !vidas)
186.                             // EL JUGADOR HA PERDIDO
187.                         alert("GAME OVER");
188.                         document.location.reload();
189.                     }
190.                     else {
191.                             // SI vidas > 0 (diferente de CERO)
   EL JUEGO CONTINUA
192.                         x = canvas.width/2;
193.                         y = canvas.height-30;
194.                         dx = 3;
195.                         dy = -3;
196.                         paletaPosX = (canvas.width-
   anchuraPaleta)/2;
197.                     }
198.                 }
199.             }
200.
201.             if(flechaDerechaPresionada && paletaPosX <
   canvas.width-anchuraPaleta) {
202.                paletaPosX += 7;
203.             }
204.             else if(flechaIzquierdaPresionada && paletaPosX > 0)
   {
205.                paletaPosX -= 7;
206.             }
207.
208.             x += dx;
209.             y += dy;
210.
211.             // ESTE ES UN SEGUNDO MÉTODO PARA REALIZAR LA
   ANIMACIÓN DEL JUEGO
212.             // LA INSTRUCCIÓN: requestAnimationFrame SE EJECUTA
   60 VECES POR SEGUNDO
213.             // Y AL EJECUTARSE LLAMA A LA FUNCIÓN ENTRE
   PARÉNTESIS
214.             // POR TANTO, dibujar SE EJECUTA 60 VECES POR SEGUNDO
215.             // GENERANDO EL CICLO DEL JUEGO
216.             requestAnimationFrame(dibujar);
217.         }
```

```
218.
219.         dibujar();
220.     </script>
221.
222.     </body>
223.     </html>
```

Gráfica 17. Código, creación de vidas y puntaje.

Al ejecutar este código se obtiene la siguiente interfaz visual:



Gráfica 18. Vidas y puntaje.

En la imagen 18 podemos observar el juego ya completado totalmente, y en el podemos observar las vidas y el puntaje que lleva el jugador durante el juego y la desaparición del mando dentro del canvas.

12 CONCLUSIONES

En conclusión, podemos observar como después de seguir una cierta cantidad de pasos pudimos llegar a nuestro objetivo principal, el cual era construir un juego clásico en 2D.

Este juego realizado a través de un código HTML asignado a JavaScript, en el cual usando las herramientas prestadas por HTML y creando y probando las funciones correctas con sus variables y problemas que surgen dentro de este código podemos llegar a tener un juego en la red virtual.

Este es un juego que nos ayuda para el aprendizaje dentro del campo de la programación tanto con el lenguaje HTML como con tantos lenguajes que existen el día de hoy en el campo de la programación.

13 BIBLIOGRAFÍA

https://developer.mozilla.org/es/docs/Games/Workflows/Famoso_juego_2D_usando_JavaScript_puro/Construye_grupo_bloques