2020

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

<JUAN GUILLERMO DUQUE MONTOYA >

SEPTIEMBRE DE 2020

UTP | Pereira

1 CONTENIDO

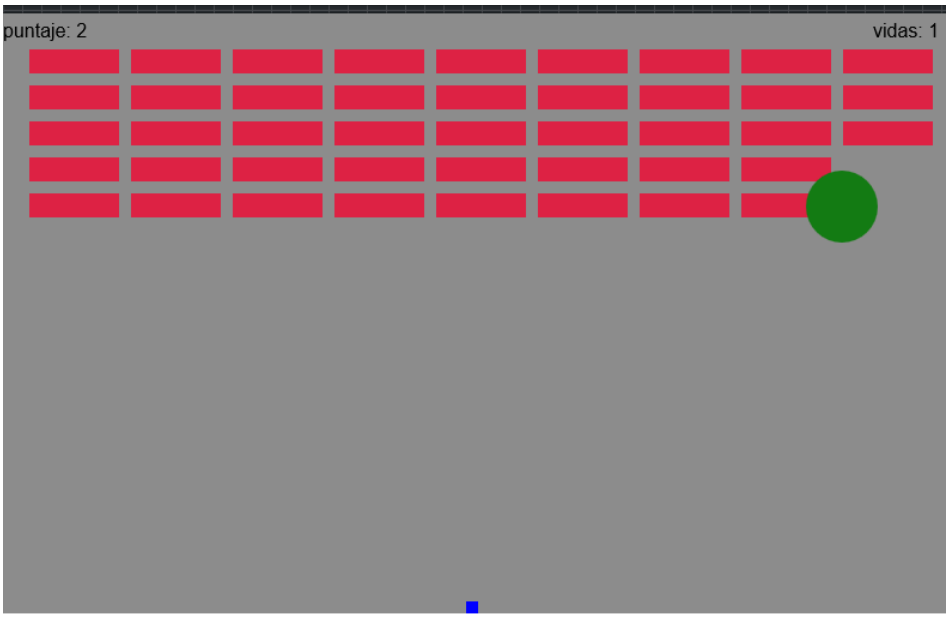
2 PRESENTACIÓN

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 fotos 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: Juan Guillermo Duque Montoya

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.

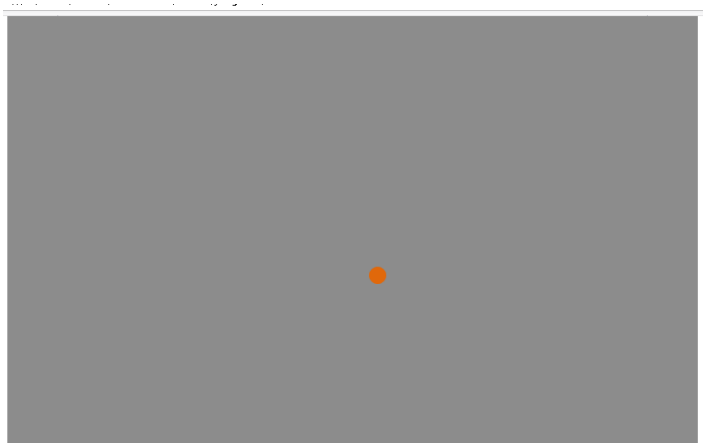
El código fuente del programa es el siguiente (para darle formato, se deben seguir las instrucciones disponibles en: https://trabajonomada.com/insertar-codigo-word/ y seguidamente utilizar el enlace: http://qbnz.com/highlighter/php_highlighter.php)

```
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: #8c8c8c;
20.                  display: block;
21.                  margin: 0 auto;
```

```
22.                 }
23.             </style>
24.
25.         </head>
26.         <body>
27.
28.         <canvas id="miCanvas" width="800" height="500"></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 -31;
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 = -2;
43.             var dy = -1;
44.
45.
46.
47.
48.             function dibujarBola() {
49.                 // Inicia el dibujo
50.                 ctx.beginPath();
51.
52.                 /* Define un círculo en las coordenadas (x, y) con
    radio 10
53.                    El ángulo va desde 0 hasta 2*PI (360 grados) */
54.                 ctx.arc(x, y, 10, 0, Math.PI*2);
55.
56.                 // Color de llenado
57.                 ctx.fillStyle = "#E0680B ";
58.
59.                 // Se llena el círculo con el color indicado
60.                 ctx.fill();
61.
62.                 // Finaliza el dibujo
63.                 ctx.closePath();
64.             }
65.
66.             /* LA FUNCIÓN dibujar REALIZA TRES TAREAS:
67.                1) Limpia el CANVAS. Inicio= (0,0) Ancho=canvas.width
    Altura=canvas.height
68.                2) Dibuja una bola en las coordenadas (x, y)
69.                3) Cambiar las coordenadas (x, y) agregando los
    valores dx, dy
```

```
70.                 Con este cambio cada vez que se dibuja la bola,
   está en una nueva posición */
71.           function dibujar() {
72.
73.               // Limpia el CANVAS
74.               ctx.clearRect(50, 0, canvas.width, canvas.height);
75.
76.               // Dibuja la bola
77.               dibujarBola();
78.
79.
80.               // Se incrementa x en el valor dx
81.               x = x + dx;
82.
83.               // Se incrementa y en el valor dy
84.               y = y + dy;
85.
86.
87.           }
88.
89.           /* EJECUTA LA FUNCIÓN dibujar CADA 10 MILISEGUNDOS
90.              Este es el mecanismo utilizado para construir un
   sistema que
91.               ejecuta acciones de manera permanente y periódica */
92.           setInterval(dibujar, 10);
93.
94.
95.       </script>
96.
97.       </body>
98.       </html>
99.
```

Al ejecutar este código se obtiene la siguiente interfaz visual:



Gráfica 2. La interfaz inicial del juego

En la gráfica 2 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 limites pemitidos 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. < cabeza >
4.      < meta charset = "utf-8" />
5.      < título > Juego 2D - lección 02 </ título >
6.      < estilo >
7.        * { relleno : 0 ; margen : 0 ; }
8.        lienzo { fondo : #
   8c8c8c ; pantalla : bloque ; margen : 0 automático ; }
9.      </ estilo >
10.       </ cabeza >
11.       < cuerpo >
12.
13.       < canvas id = "miCanvas" width = "800" height = "500" >
   </ canvas >
14.
15.       < guión >
16.           var lienzo = documento. getElementById ( "miCanvas" ) ;
17.           var ctx = lienzo. getContext ( "2d" ) ;
18.
19.           / * Se agrega la variable ballRadius, la cual define el
   tamaño de la bola * /
20.           var ballRadius = 30 ; // <--------------------
21.
22.           var x = lienzo. ancho / 2 ;
23.           var y = lienzo. altura - 30 ;
24.           var dx = 10 ;
25.           var dy = - 20 ;
26.
27.           function dibujarBola ( ) {
28.               ctx. beginPath ( ) ;
29.
30.               / * En lugar de un número fijo, se coloca la variable
   ballRadius * /
```

```
31.             ctx. arco ( x , y , ballRadius , 0 , Math . PI * 2 )
   ; // <----------------
32.
33.             ctx. fillStyle = "# 245D1A" ;
34.             ctx. llenar ( ) ;
35.             ctx. closePath ( ) ;
36.         }
37.
38.         function dibujar ( ) {
39.             ctx. clearRect ( 0 , 0 , lienzo. ancho , lienzo. alto
    ) ;
40.             dibujarBola ( ) ;
41.
42.             / * IMPORTANTE:
43.
44.                EL OPERADOR || es el operador lógico O
45.                Este operador se utiliza para indicar la condición
   de conjunción
46.                SI SE CUMPLE UNA CONDICIÓN, O SE CUMPLE OTRA
   CONDICIÓN, ENTONCES
47.                SE CUMPLE LA CONDICIÓN
48.
49.                EL OPERADOR && es el oerador lógico AND
50.                Este operador se utiliza para indicar la condición
   de disyunción
51.                SI SE CUMPLE UNA CONDICIÓN, Y SE CUMPLE OTRA
   CONDICIÓN (simultánea), ENTONCES
52.                SE CUMPLE LA CONDICIÓN
53.
54.             * /
55.
56.             / * DESPUÉS DE DIBUJAR LA BOLA, SE DEBEN CAMBIAR LAS
   COORDENADAS
57.                EN LA lección 01 NO SE TENÍA CONTROL SOBRE LOS
   LÍMITES DE LA CAJA
58.                ----------------------------------------------------
   ---------------
59.                SI x + dx ES MAYOR AL ANCHO DEL CANVAS O MENOR AL
   TAMAÑO DEL
60.                RADIO DE LA BOLA (caso en el cual se encuentra
   hacia la izquierda)
61.                SE CAMBIA LA DIRECCIÓN DE AVANCE HORIZONTAL.
62.                ESTO SE LOGRA CAMBIANDO EL SIGNO DE LA VARIABLE dx
63.                ESTO HACE QUE SE CAMBIE EL SENTIDO DEL MOVIMIENTO
   HORIZONTAL * /
64.             if ( x + dx > lienzo. ancho -
    ballRadius || x + dx < ballRadius ) { // <----------------
65.                 dx = dx * ( - 1 ) ;
66.             }
67.
68.             / * SI y + dy ES MAYOR A LA ALTURA DEL CANVAS O MENOR
   AL TAMAÑO DEL
69.                RADIO DE LA BOLA, SE CAMBIA LA DIRECCIÓN DEL
   AVANCE VERTICAL.
```

```
70.                 ESTO SE LOGRA CAMBIANDO EL SIGNO DE LA VARIABLE dy
71.                 ESTE CAMBIO EN dy HACE QUE SE MUEVA VERTICALMENTE
   EN SENTIDO
72.                 OPUESTO * /
73.             if ( y + dy > lienzo. altura -
    ballRadius || y + dy < ballRadius ) { // <-----------------
74.                  dy = - dy ;
75.             }
76.
77.             / * AQUÍ SE CAMBIA LA POSICIÓN DE LA BOLA. SE TOMA EN
   CUENTA LAS
78.                 MODIFICACIONES A dx y dy, EN CASO DE QUE SE
   HUBIERAN PRODUCIDO * /
79.             x + = dx ; // EQUIVALE A: x = x + dx; <--------------
   --
80.             y + = dy ; // EQUIVALE A: y = y + dy; <--------------
   --
81.          }
82.
83.           setInterval ( dibujar , 10 ) ;
84.      </ script >
85.
86.      </ cuerpo >
87.      </ html >
```

Al ejecutar este código se obtiene la siguiente interfaz visual:

En la figura 3 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 PALA Y EL TECLAD

El paso numero 3 consiste en crear la paleta en la cual la bola rebotara, la cual estará situada en la parte inferior del juego y sera 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. < cabeza >
4.     < meta charset = "utf-8" />
5.     < title > Juego 2D - # 03 - Paleta y Control por
   Teclado </ title >
6.     < estilo >
7.         * { relleno : 0 ; margen : 0 ; }
8.         lienzo { fondo : #
   8c8c8c ; pantalla : bloque ; margen : 0 automático ;
    }
9.     </ estilo >
10.    </ cabeza >
11.    < cuerpo >
12.
13.    < canvas
   id = "miCanvas" width = "800" height = "500" >
   </ canvas >
14.
15.    < guión >
16.        var lienzo = documento. getElementById ( "m
   iCanvas" ) ;
17.        var ctx = lienzo. getContext ( "2d" ) ;
18.
19.        / * Variables básicas:
20.
21.           radioBola: radio de la esfera
```

```
22.           x: columna en la que se encuentra
  situada la bola
23.           y: fila en la que se encuentra situada
  la bola
24.           dx: desplazamiento horizontal de la bola
25.           dy: desplazamiento vertical de la bola
26.
27.       NOTAS: originalmente, la bola está en lada
  por la flecha izquierda y
28.               la flecha derecha del teclado (luego
  será controlador por el ratón)
29.
30.               alturaPaleta: define la altura de la
  paleta en pixeles
31.               anchuraPaleta: define la anchura de
  la paleta
32.
33.               NOTA: Estos dos valores determinan
  el tamaño de la paleta
34.                     La paleta se encuentra situada
  en la base de la pantalla
35.                     Para calcular la posición en X
  de la paleta, se debe tomar
36.                     el ancho del CANVAS, restarle
  la anchura de la paleta, y
37.                     el espacio que sobre debe
  dividirse entre dos
38.                     Esto garantiza que
  inicialmente la paleta estará centrada
39.                     en la base de la pantalla
40.
41.               Al inicio del juego, aún no se ha
  presionado ninguna de las
42.               flechas. Esta es la razón por la
  cual se define dos variables que
43.               "recuerdan" cual de las flechas se
  ha presionado, pero que
44.               están puestas a: falso, indicando el
  estado inicial
45.               Cuando se pulse cualquiera de las
  dos flechas, su valor será:
```

```
46.                 true (verdadero), y este valor
  permitira establecer en qué
47.                 dirección se debe mover la paleta
  (dentro del ciclo del juego)
48.                 Las variables hijo:
49.
50.                 flechaDerechaPulsada
51.                 flechaIzquierdaPulsada
52.
53.                 NOTA: Desde ahora debe tomarse en
  cuenta que cuando se pulse
54.                       cualquiera de las dos flechas,
  solamente se hará un
55.                       desplazamiento de la paleta a
  la izquierda o hacia la
56.                       derecha. Si se mantiene
  pulsada la tecla, la paleta se
57.                       continuará desplazando, hasta
  alcanzar el extremo derecho
58.                       o izquierdo de la pantalla del
  juego
59.         * /
60.
61.         var alturaPaleta = 20 ; // <---------------
62.         var anchuraPaleta = 200 ; // <-------------
  --
63.         var paletaPosX = ( lienzo. ancho -
   anchuraPaleta ) / 2 ; // <------------------
64.
65.         var flechaDerechaPulsada = false ; // <----
  -------------
66.         var flechaIzquierdaPulsada = false ; // <--
  ---------------
67.
68.         / * La instruccion: addEventListener, se
  utiliza para crear un
69.             mecanismo de respuesta ante eventos que
  se produce en el juego
70.
71.             addEventListener "agregar un mecanismo
  que deteca y recibe eventos"
```

```
72.
73.             addEventListener recibe tres parámetros:
74.
75.             1) El evento que se va a detectar
76.             2) El nombre que le asignamos a la
  función que responde ante el evento
77.             3) Valor true o false que determina la
  reacción ante el evento
78.
79.             Los dos primeros parámetros son fáciles
  de entender. Pero el tercero
80.             requiere de una explicación adicional:
81.
82.             Para entender el tercer parámetro,
  primero hemos de saber lo que es
83.             el flujo de eventos.
84.         Cuando hacemos clic en el botón no sólo lo
  estamos haciendo sobre él,
85.             sino sobre los elementos que lo
  contienen en el árbol de la página,
86.             es decir, hemos hecho clic, además,
  sobre el elemento <body> y sobre
87.             el elemento <div>. Sí sólo hay una
  función asignada a una escucha
88.             para el botón, no hay mayor problema,
  pero si además hay una
89.             eschucha para el body y otra para el
  div,
90.             ¿Cuál es el orden en que se deben lanzar
  las tres funciones?
91.
92.                 Para contestar a esa pregunta existe
  un modelo de comportamiento,
93.                 el flujo de eventos. Según éste,
  cuando se hace clic sobre un
94.                 elemento, el evento se propaga en
  dos fases, una que es la
95.                 captura —el evento comienza en el
  nivel superior del documento
96.                 y recorre los elementos de padres a
  hijos— y la otra la burbuja
```

```
97.               —El orden inverso, ascendiendo de
  hijos a padres—.
98.
99.               Así, el orden por defecto de
  lanzamiento de las funciones
100.              de escucha, sería: primero la
  función de escuch de body,
101.              luego la función de escucha de div,
  y por último la función
102.              de escucha de button.
103.
104.              Una vez visto esto, podemos
  comprender el tercer parámetro de addEventListener,
  que lo que hace es permitirnos elegir el orden de
  propagación:
105.
106.          true: El orden de propagación para el
  ejemplo sería, por tanto,
107.          botón-div-cuerpo
108.
109.          falso: La propagación seguiría el modelo
  burbuja.
110.          Así, el orden sería button-div-body.
111.
112.          NOTA: omo en nuestro ejemplo utilizamos
  "falso", estamos
113.                eaccionando primero ante el evento
  sobre las teclas,
114.                 posteriormente sobre los eventos
  asociados al CANVAS.
115.                ste es el mecanismo más usual,
  pero se utilizará "true"
116.                n las situaciones que lo requieran
117.       * /
118.       documento. addEventListener ( "keydown" , m
  anejadorTeclaPresionada , false ) ; // <---------
119.       documento. addEventListener ( "keyup" , man
  ejadorTeclaLiberada , false ) ; // <--------
120.
121.       // Función que maneja tecla presionada
```

```
122.        función manejadorTeclaPresionada ( e ) { //
  <----------------
123.            if ( e. keyCode == 39 ) {
124.                    / * e: Es el evento que se
  produce, en este caso
125.                                            tecla
  presionada. La propiedad: keyCode permite
126.                                            descubrir de
  qué tecla se trata. Si el código es 39,
127.                                            se ha
  presionado la flecha derecha. En este caso
128.                                            se coloca la
  variable: flechaDerechaPulsada a true
129.                                    * /
130.                flechaDerechaPulsada = true ;
131.            }
132.            else if ( p. keyCode == 37 ) {
133.                    / * e: Es el evento que se
  produce, en este caso
134.                                            tecla
  presionada. La propiedad: keyCode permite
135.                                            descubrir de
  qué tecla se trata. Si el código es 37,
136.                                            se ha
  presionado la flecha izquierda. En este caso
137.                                            se coloca la
  variable: flechaIquierdaPulsada a true
138.                                    * /
139.                flechaIzquierdaPulsada = verdadero
  ;
140.            }
141.        }
142.
143.        // Función que maneja tecla liberada
144.        función manejadorTeclaLiberada ( e ) { //
  <--------------------
145.            if ( e. keyCode == 39 ) {
146.                    / * Si la tecla liberada es la
  39, se ha dejado de
147.                        presionar la flecha
  derecha. En este caso, la variable
```

```
148.                         se pone en: falso
149.                     * /
150.                 flechaDerechaPulsada = falso ;
151.             }
152.             else if ( e. keyCode == 37 ) { // <----
  -------------
153.                     / * Si la tecla liberada es la
  37, se ha dejado de
154.                         presionar la flecha
  izquierda. En este caso, la variable
155.                         se pone en: falso
156.                     * /
157.                 flechaIzquierdaPulsada = falso ;
158.             }
159.         }
160.
161.         // Dibuja la bola. Código explicado en
  programas anteriores
162.         function dibujarBola ( ) {
163.             ctx. beginPath ( ) ;
164.             ctx. arco ( x , y , radioBola , 0 , Mat
  h . PI * 2 ) ;
165.             ctx. fillStyle = "# ff0000" ;
166.             ctx. llenar ( ) ;
167.             ctx. closePath ( ) ;
168.         }
169.
170.         function dibujarPaleta ( ) { // <----------
  ----------
171.             // Se inicia el dibujo de la paleta
172.             ctx. beginPath ( ) ;
173.             / * Se crea un rectángulo utilizando la
  posición en X
174.                 El valor de Y está en la base de la
  pantalla menos la
175.                 altura de la paleta
176.                 Y a continuación se indica la
  anchura y la altura de la paleta
177.                     * /
178.             ctx. rect ( paletaPosX , lienzo. altura
   - alturaPaleta , anchuraPaleta , alturaPaleta ) ;
```

```
179.            ctx. fillStyle = "# E0680B" ;
180.            ctx. llenar ( ) ;
181.            // Se "cierra" la paleta, terminando su
  dibujo
182.            ctx. closePath ( ) ;
183.        }
184.
185.        // Función principal. A partir de aquí se
  origina el proceso
186.        // general del juego
187.        function dibujar ( ) {
188.            ctx. clearRect ( 50 , 0 , lienzo. ancho
   , lienzo. alto ) ;
189.
190.            // En primer lugar, dibuja la bola
191.            dibujarBola ( ) ;
192.
193.            // Seguidamente, dibuja la paleta
194.            dibujarPaleta ( ) ; // <--------------
  -----
195.
196.            / * IMPORTANTE:
197.
198.               EL OPERADOR || es el operador lógico
  O
199.               Este operador se utiliza para
  indicar la condición de conjunción
200.               SI SE CUMPLE UNA CONDICIÓN, O SE
  CUMPLE OTRA CONDICIÓN, ENTONCES
201.               SE CUMPLE LA CONDICIÓN
202.
203.               EL OPERADOR && es el oerador lógico
  AND
204.               Este operador se utiliza para
  indicar la condición de disyunción
205.               SI SE CUMPLE UNA CONDICIÓN, Y SE
  CUMPLE OTRA CONDICIÓN (simultánea), ENTONCES
206.               SE CUMPLE LA CONDICIÓN
207.
208.            * /
209.
```

```
210.            // Aquí se controla los límites a los
  que puede llegar la bola
211.            // En caso de intentar sobrepasar
  dichos límites, se cambia
212.            // el sentido del movimiento
213.            // Este código se describe en el
  programa anterior
214.            if ( x + dx > lienzo. ancho -
   radioBola || x + dx < radioBola ) { // <----------
  ---
215.                dx = - dx ;
216.            }
217.            if ( y + dy > lienzo. altura -
   radioBola || y + dy < radioBola ) { // <----------
  ---
218.                dy = - dy ;
219.            }
220.
221.            / * Si se ha pulsado la flecha derecha,
  y la paleta aún puede
222.              desplazarse hacia la derecha sin que
  se sobrepase el límite de la
223.              pantalla, entonces se procede a
  cambiar su posición
224.              En este caso, la función:
  dibujarPaleta (la cual se ejecuta de
225.              manera cíclica) redibujará la paleta
  en la nueva posición
226.            * /
227.            if ( flechaDerechaPulsada && paletaPosX
   < lienzo. ancho - anchuraPaleta ) { // <--------
228.                  // Se desplaza la paleta hacia
  la derecha
229.                  // Aquí, paletaPosX + = 7
  equivale a: paletaPosX = paletaPosX + 7
230.                paletaPosX + = 7 ;
231.            }
232.            else if ( flechaIzquierdaPulsada && pal
  etaPosX > 0 ) { // <------------
233.                  // Se desplaza la paleta hacia
  la izquierda
```

```
234.                    // Aquí, paletaPosX - = 7
  equivale a: paletaPosX = paletaPosX - 7
235.                paletaPosX - = 7 ;
236.            }
237.
238.            x + = dx ;
239.            y + = dy ;
240.        }
241.
242.        / * Con esta instrucción se crea un
  ciclo. Cada 10 milisegundos se
243.           ejecuta la funcion: dibujar (). Esto
  genera el ciclo que permitirá
244.           actualizar el juego, detectar eventos y
  cambiar el estado
245.
246.    Al ejecutar este código se obtiene la siguiente interfaz visual:
```

En la figura 4 podemos observar la bola y la paleta en la parte inferior del juego creadas anteriormente en la parte numero 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 codigo 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 lo bola toque la parte inferior hace que se detenga el ciclo de animación del juego y se pierda.

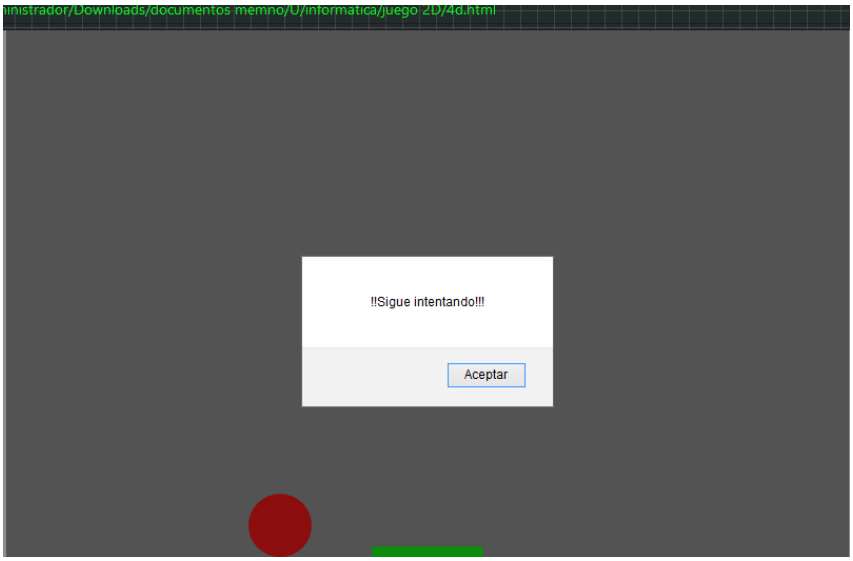
```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.     <meta charset="utf-8" />
5.     <title>Juego 2D - #04 - Game Over</title>
6.     <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
9.
10.      <canvas id="miCanvas" width="480" 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;
```

```
            }
            if(y + dy < radioBola) {
                dy = -dy;
            }

            /* Si y + dy alcanza la frontera inferior de la pantalla
               (y + dy > canvas.height - radioBola)
               existe la posibilidad de que el jugador pierda el juego
               Para ello debe evaluarse una segunda opción:
               La variable x determina la posición de la bola
               Lo que debe hacerse es mirar si x está DENTRO de la palata:
               (x > paletaPosX && x < paletaPosX + anchuraPaleta)
                       ----------------------------------------------
                       Si x está dentro de la paleta, todo va bien y se incrementa y
                       ----------------------------------------------
                       Si x NO ESTÁ dentro de la paleta (else), la bola ha llegado
                       a la frontera inferior, y no encuentra la paleta en su camino
                       En este caso, SE DETIENE EL CICLO DE ANIMACIÓN, y se genera
                       un ALERT indicando que el jugador ha perdido (GAME OVER)
            */
            else if(y + dy > canvas.height-radioBola) {
                if(x > paletaPosX && x < paletaPosX + anchuraPaleta) {
                    dy = -dy;
                }
                else {
                    clearInterval(juego);
                    alert("GAME OVER");
                    document.location.reload();
                }
            }

            if(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
                paletaPosX += 7;
            }
            else if(flechaIzquierdaPresionada && paletaPosX > 0) {
                paletaPosX -= 7;
            }

            x += dx;
            y += dy;
        }
```

```
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>
```

Al ejecutar este código se obtiene la siguiente interfaz visual:



En la figura 5 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

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 sgte funcion: 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.

```
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;
```

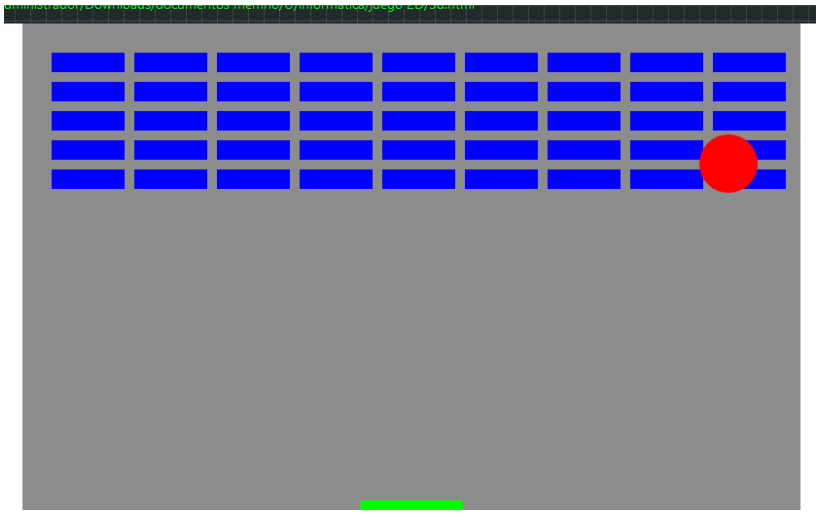
```
24.          var flechaIzquierdaPresionada = false;
25.
26.          /* NUEVAS VARIABLES asociadas a los ladrillos
27.          */
28.          var nroFilasLadrillos = 5;
29.          var nroColumnasLadrillos = 3;
30.          var anchoLadrillo = 75;
31.          var alturaLadrillo = 20;
32.          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>
```

Al ejecutar este código se obtiene la siguiente interfaz visual:



En la figura 6 podemos observar la creación de la pared de ladrillos dentro del campo del 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.

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 hara 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;
```

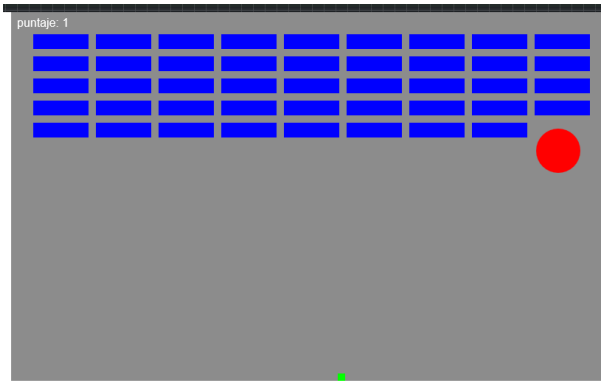
```
32.          var alturaLadrillo = 20;
33.          var rellenoLadrillo = 10;
34.          var vacioSuperiorLadrillo = 30;
35.          var vacioIzquierdoLadrillo = 30;
36.
37.          var ladrillos = [];
38.          for(var c=0; c<nroColumnasLadrillos; c++) {
39.             ladrillos[c] = [];
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:



En la figura 7 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 numero de ladrillos haciendo que el juego se gane.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Juego 2D - #07 - Control de juego ganado</title>
    <!-- EN ESTE EJEMPLO SE CAMBIA LA ANCHURA DE LA PALETA
         ESTO ES CLAVE PARA PERMITIR QUE EL JUEGO SEA AUTOMÁTICO
         Y SE PUEDA VERIFICAR EL OBJETIVO DEL JUEGO Y EL JUGADOR
GANE -->
```

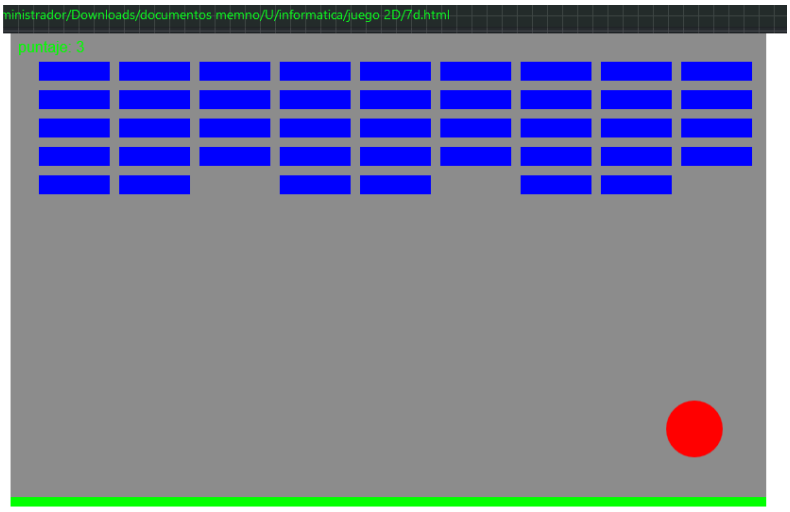
```
9.      <style>* { padding: 0; margin: 0; } canvas { background: #eee;
   display: block; margin: 0 auto; }</style>
10.     </head>
11.     <body>
12.
13.     <canvas id="miCanvas" width="480" height="320"></canvas>
14.
15.     <script>
16.         var canvas = document.getElementById("miCanvas");
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>
```

Al ejecutar este código se obtiene la siguiente interfaz visual:



En la figura 8 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.

```
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);
```

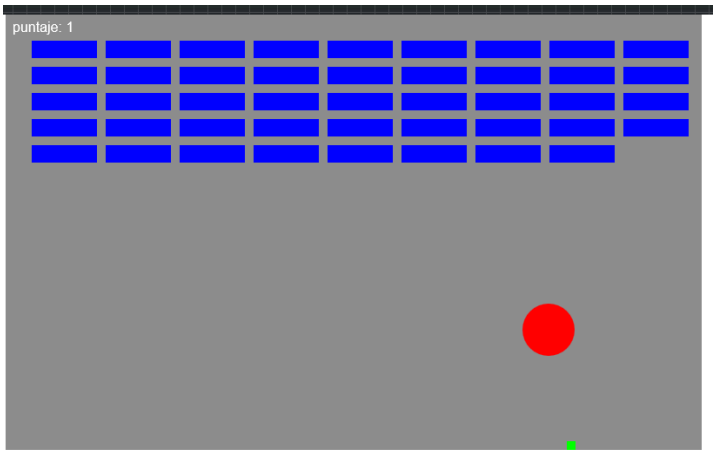
```
126.             ctx.fillStyle = "#0095DD";
127.             ctx.fill();
128.             ctx.closePath();
129.         }
130.         function dibujarPaleta() {
131.             ctx.beginPath();
132.             ctx.rect(paletaPosX, canvas.height-alturaPaleta,
    anchuraPaleta, alturaPaleta);
133.             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>
```

Al ejecutar este codigo se obtiene la siguiente interfaz visual:

En la figura 9 se puede observar como la paleta es controlada de derecha a izquierda mediante el uso del mouse

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.


puntaje: 1

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 y que ha perdido.

```
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;
```

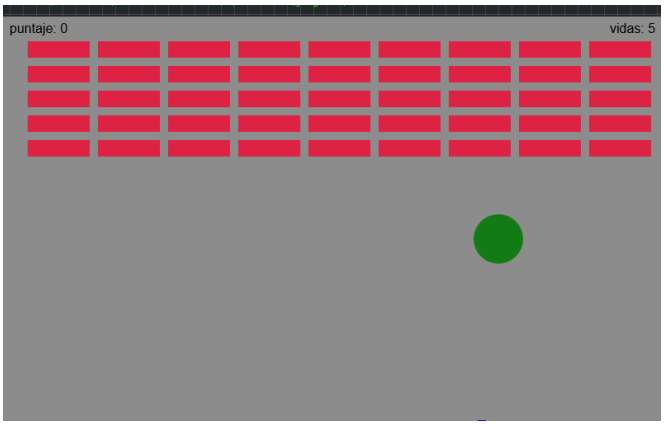
```
39.
40.          var puntaje = 0;
41.
42.          // ESTA INSTRUCCIÓN CONTROLA EL NÚMERO DE VIDAS DEL
   JUGADOR
43.          // CUANDO LA INSTRUCCIÓN vidas DISMINUYE A CERO, EL
   JUGADOR PIERDE,
44.          // PUESTO QUE HA PERDIDO TRES VECES
45.          var vidas = 3;
46.
47.          // ESTA 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) {
```

```
                        var ladrilloX = (f*(anchuraLadrillo+rellenoLadrillo))+vacioIzquierdoLadrillo;
                        var ladrilloY = (c*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLadrillo;
                        ladrillos[c][f].x = ladrilloX;
                        ladrillos[c][f].y = ladrilloY;
                        ctx.beginPath();
                        ctx.rect(ladrilloX, ladrilloY, anchuraLadrillo, alturaLadrillo);
                        ctx.fillStyle = colorLadrillo;
                        ctx.fill();
                        ctx.closePath();
                    }
                }
            }
        }

        function dibujarPuntaje() {
            ctx.font = "16px Arial";
            ctx.fillStyle = colorTexto;
            ctx.fillText("puntaje: "+puntaje, 8, 20);
        }

        function dibujarVidas() {
            ctx.font = "16px Arial";
            ctx.fillStyle = colorTexto;
            // SE MUESTRA EL NÚMERO DE VIDAS DISPONIBLES
            ctx.fillText("vidas: "+vidas, canvas.width-65, 20);
        }

        function dibujar() {
            ctx.clearRect(0, 0, canvas.width, canvas.height);
            dibujarLadrillos();
            dibujarBola();
            dibujarPaleta();
            dibujarPuntaje();
            dibujarVidas();
            detectarColision();

            if(x + dx > canvas.width-bolaRadio || x + dx < bolaRadio) {
                dx = -dx;
            }
            if(y + dy < bolaRadio) {
                dy = -dy;
            }
            else if(y + dy > canvas.height-bolaRadio) {
                 if(x > paletaPosX && x < paletaPosX + anchuraPaleta) {
                    dy = -dy;
                 }
                 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>
```

Al ejecutar este codigo se obtiene la siguiente interfaz visual:



En la imagen 10 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 el cual era construir un juego en 2D.

Este juego realizado a través de un codigo html asignado a JavaSvript, 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 codigo 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 dia 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