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

UTP | Pereira

**juan camilo rodríguez libreros**

septiembre DE 2020

2020

# CONTENIDO

[1 CONTENIDO 1](#_Toc42134755)

[2 PRESENTACIÓN 2](#_Toc42134756)

[3 FASE 1: Dibujar y mover una bola 3](#_Toc42134757)

[4 FASE 2: Rebotando en las paredes 7](#_Toc42134758)

[5 FASE 3: Control de la pala y el teclado 1](#_Toc42134759)1

[6 FASE 4: Fin del juego 2](#_Toc42134760)0

[7 FASE 5: Muro de ladrillos 2](#_Toc42134761)5

[8 FASE 6: Detección de colisiones](#_Toc42134762) 30

[9 FASE 7: Contar puntos y ganar 3](#_Toc42134763)6

[10 FASE 8: Controlando el ratón 41](#_Toc42134764)

[11 FASE 9: Finalizando el juego 47](#_Toc42134765)

[12 CONCLUSIONES 53](#_Toc42134766)

[13 BIBLIOGRAFÍA 53](#_Toc42134767)

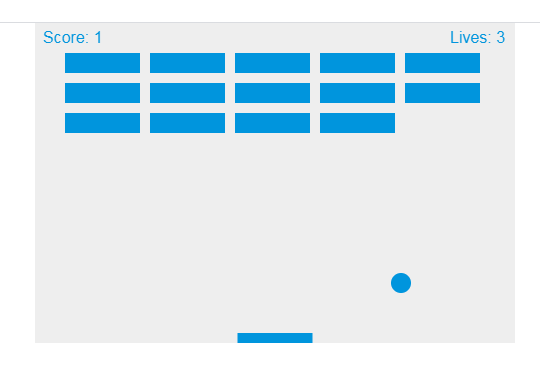
# PRESENTACIÓN

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 CAMILO RODRÍGUEZ LIBREROS>**

# FASE 1: Dibujar y mover 3 bolas

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, este tiene un tamaño de 600x400.

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 encuentran las bolas y los valores de incremento para modificar su posición.

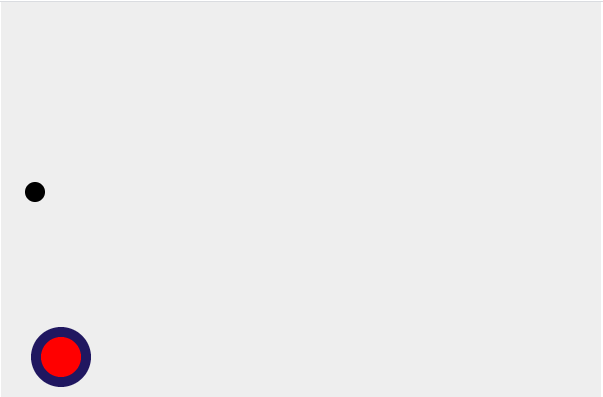
Se definen 5 funciones importantes. La primera de ellas, dibujarBola(), se encarga de dibujar sobre la pantalla una bola con el color y tamaños indicados en los estilos. La segunda función se denomina dibujarbola2(), esta dibuja una segunda bola que va en conjunto con la anterior, esta tiene un color y tamaño diferente para que se diferencie la una de la otra ya que van en el mismo rumbo, esto hace que parezca una sola bola con un estilo diferente. La tercer función se denomina dibujarbola3() hace lo mismo que las funciones anteriores, a esta se le crearon coordenadas diferentes denominadas A, B para que tome un rumbo diferente de las otras 2. La cuarta función se denomina dibujar(), y es la encargada de limpiar el CANVAS, dibujar las 3 bolas y cambiar los valores de las coordenadas. Finalmente, la función setInterval(dibujar, 150), llama a la función dibujar cada 150 milisegundos, esto se hace con el fin de que el movimiento sea un poco más lento y se pueda apreciar mejor.

El código fuente del programa es el siguiente:

1. <!DOCTYPE html>
2. <[**html**](http://december.com/html/4/element/html.html)>
3. <[**head**](http://december.com/html/4/element/head.html)>
4. <[**meta**](http://december.com/html/4/element/meta.html) charset="utf-8" />
5. <[**title**](http://december.com/html/4/element/title.html)>Juego 2D: JavaScript - 01</[**title**](http://december.com/html/4/element/title.html)>
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**](http://december.com/html/4/element/style.html)>
14. \* {
15. padding: 0;
16. margin: 0;
17. }
18. canvas {
19. background: #eeeeee;
20. display: block;
21. margin: 0 auto;
22. }
23. </[**style**](http://december.com/html/4/element/style.html)>
25. </[**head**](http://december.com/html/4/element/head.html)>
26. <[**body**](http://december.com/html/4/element/body.html)>
28. <[**canvas**](http://december.com/html/4/element/canvas.html) id="miCanvas" width="600" height="400"></[**canvas**](http://december.com/html/4/element/canvas.html)>
30. <[**script**](http://december.com/html/4/element/script.html)>
32. ///////////////////////////////////////////////////////////////////////////
33. var canvas = document.getElementById("miCanvas");
35. // -----------------------------------------------------------------------
36. var ctx = canvas.getContext("2d");
37. // -----------------------------------------------------------------------
38. var a = canvas.width/150
40. var b = canvas.height-30
41. // Coloca x en la mitad del ancho deL CANVAS
42. var x = canvas.width/20;
44. // Coloca y en la mitad de la altura del CANVAS (restando 30 a dicho valor)
45. var y = canvas.height-30;
47. /\* DEFINE LOS INCREMENTOS EN X,A y en Y,B. El valor dy y db es negativo
48. para que inicialmente el movimiento de la bola sea hacia arriba \*/
49. var dx = 10;
50. var dy = -5;
51. var da = 10
52. var db = -60
54. function dibujarBola() {
55. // Inicia el dibujo
56. ctx.beginPath();
58. /\* Define un círculo en las coordenadas (x, y) con radio 30
59. El ángulo va desde 0 hasta 2\*PI (360 grados) \*/
60. ctx.arc(x, y, 30, 0, Math.PI\*2);
62. // Color de llenado
63. ctx.fillStyle = "#201760";
65. // Se llena el círculo con el color indicado
66. ctx.fill();
68. // Finaliza el dibujo
69. ctx.closePath();
70. }
72. //esta bola va junto con la bola 1 dandole una apariencia distinta
74. function dibujarbola2() {
76. ctx.beginPath();
77. ctx.arc(x, y, 20, 0, Math.PI\*2);
78. ctx.fillStyle = "#ff0000";
79. ctx.fill();
80. ctx.closePath();
81. }
83. //esta otra toma un rumbo distinto
84. function dibujarbola3() {
86. ctx.beginPath();
87. ctx.arc(a, b, 10, 0, Math.PI\*2);
88. ctx.fillStyle = "#000000";
89. ctx.fill();
90. ctx.closePath();
91. }

94. /\* LA FUNCIÓN dibujar REALIZA TRES TAREAS:
95. 1) Limpia el CANVAS. Inicio= (0,0) Ancho=canvas.width Altura=canvas.height
96. 2) Dibuja las bolas 1 y 2 en las coordenadas (x, y) y la bola 3 en las coordenadas (a, b)
97. 3) Cambiar las coordenadas (x, y),(a, b) agregando los valores dx, dy, da, db.
98. Con este cambio cada vez que se dibuja la bola, está en una nueva posición \*/
99. function dibujar() {
101. // Limpia el CANVAS
102. ctx.clearRect(0, 0, canvas.width, canvas.height);
104. // Dibuja las bolas
105. dibujarBola();
107. dibujarbola2()
109. dibujarbola3()
111. // se incrementa a en el valor da
112. a = a + da;
114. // se incrementa b en el valor db
115. b = b + db;
117. // Se incrementa x en el valor dx
118. x = x + dx;
120. // Se incrementa y en el valor dy
121. y = y + dy;
122. }
124. /\* EJECUTA LA FUNCIÓN dibujar CADA 150 MILISEGUNDOS
125. Este es el mecanismo utilizado para construir un sistema que
126. ejecuta acciones de manera permanente y periódica \*/
127. setInterval(dibujar, 1000);
129. </[**script**](http://december.com/html/4/element/script.html)>
131. </[**body**](http://december.com/html/4/element/body.html)>
132. </[**html**](http://december.com/html/4/element/html.html)>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Gráfica 2. La interfaz fase 1**

En la gráfica 2 se aprecia los dibujos de las 3 bolas, y la secuencia de movimiento a partir de los incrementos en X, Y y en A, B (para que la bola 3 tome un rumbo distinto) que fueron definidos.

En el siguiente apartado se explicará la siguiente fase del juego.

# FASE 2: Rebotando en las paredes

El segundo paso consiste en elaborar los límites a los que la bola va a llegar y va a rebotar 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>
3. <html>



8. <head>

11. <meta charset="utf-8" />



16. <title>Juego 2D - lección 02</title>



21. <style>

24. \* { padding: 0; margin: 0; }

27. canvas { background: #0BE9FF; display: block; margin: 0 auto; }

30. </style>


34. </head>
36. <body>
38. <canvas id="miCanvas" width="600" height="400"> </canvas>

41. <script>



46. **var** canvas = document.getElementById("miCanvas");
48. **var** ctx = canvas.getContext("2d");



53. /\* Se agrega la variable ballRadius, la cual define el tamaño de la bola \*/
54. **var** ballRadius = 20; // <--------------------
56. **var** x = canvas.width / 2;
58. **var** y = canvas.height - 30;


62. **var** dx = 2;
64. **var** dy = -2;

67. **function** dibujarBola() {

70. ctx.beginPath();


74. /\* En lugar de un número fijo, se coloca la variable ballRadius \*/
75. ctx.arc(x, y, ballRadius, 0, Math.PI\*2); // <----------------
77. ctx.fillStyle = "#FF420B";
79. ctx.fill();


83. ctx.closePath();

86. }
88. **function** dibujar() {

91. ctx.clearRect(0, 0, canvas.width, canvas.height);

94. dibujarBola();
96. /\* IMPORTANTE:
98. EL OPERADOR || es el operador lógico OR
99. Este operador se utiliza para indicar la condición de conjunción
100. SI SE CUMPLE UNA CONDICIÓN, O SE CUMPLE OTRA CONDICIÓN, ENTONCES
101. SE CUMPLE LA CONDICIÓN
103. EL OPERADOR && es el oerador lógico AND
104. Este operador se utiliza para indicar la condicion de disyunción
105. SI SE CUMPLE UNA CONDICIÓN, Y SE CUMPLE OTRA CONDICIÓN (simultánea), ENTONCES
106. SE CUMPLE LA CONDICIÓN
108. \*/
110. /\* DESPUÉS DE DIBUJAR LA BOLA, SE DEBEN CAMBIAR LAS COORDENADAS
111. EN LA lección 01 NO SE TENÍA CONTROL SOBRE LOS LÍMITES DE LA CAJA
112. -----------------------------------------------------------------
113. SI x + dx ES MAYOR AL ANCHO DEL CANVAS O MENOR AL TAMAÑO DEL
114. RADIO DE LA BOLA (caso en el cual se encuentra hacia la izquierda)
115. SE CAMBIA LA DIRECCIÓN DE AVANCE HORIZONTAL.
116. ESTO SE LOGRA CAMBIANDO EL SIGNO DE LA VARIABLE dx
117. ESTO HACE QUE SE CAMBIE EL SENTIDO DEL MOVIMIENTO HORIZONTAL \*/
118. **if** (x + dx > canvas.width-ballRadius || x + dx < ballRadius) { // <-----------------
119. dx = dx \* (-1);
120. }
122. /\* SI y + dy ES MAYOR A LA ALTURA DEL CANVAS O MENOR AL TAMAÑO DEL
123. RADIO DE LA BOLA, SE CAMBIA LA DIRECCIÓN DEL AVANCE VERTICAL.
124. ESTO SE LOGRA CAMBIANDO EL SIGNO DE LA VARIABLE dy
125. ESTE CAMBIO EN dy HACE QUE SE MUEVA VERTICALMENTE EN SENTIDO
126. OPUESTO \*/
127. **if**(y + dy > canvas.height - ballRadius || y + dy < ballRadius) { // <-----------------
128. dy = dy \* (-1);
129. }
131. /\* AQUÍ SE CAMBIA LA POSICIÓN DE LA BOLA. SE TOMA EN CUENTA LAS
132. MODIFICACIONES A dx y dy, EN CASO DE QUE SE HUBIERAN PRODUCIDO \*/
133. x = x + dx; // EQUIVALE A: x = x + dx; <----------------
134. y = y + dy; // EQUIVALE A: y = y + dy; <----------------
135. }
137. setInterval(dibujar, 10);
139. </script>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 3. Interfaz fase 2**

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.

# FASE 3: Control de la pala y el teclado

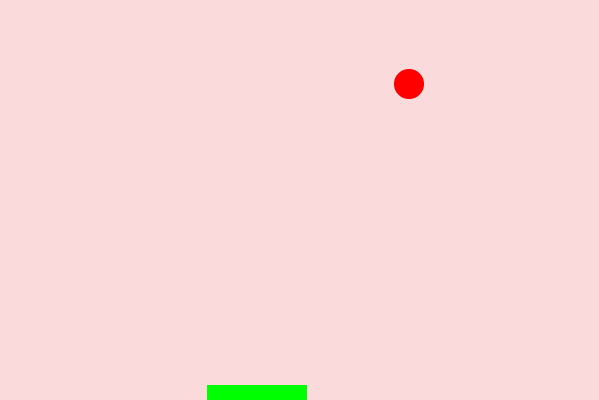
El paso numero 3 consiste en crear la paleta en la cual la bola va a rebotar, la cual estará situada en la parte inferior del juego y será controlada por ambas flechas del teclado.

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>
7. \* { padding: 0; margin: 0; }
8. canvas { background: #FADADA; display: block; margin: 0 auto; }
9. </style>
10. </head>
11. <body>
13. <canvas id="miCanvas" width="600" height="400"></canvas>
15. <script>
16. **var** canvas = document.getElementById("miCanvas");
17. **var** ctx = canvas.getContext("2d");

20. **var** radioBola = 15; // <-----------------
22. **var** x = canvas.width/2;
23. **var** y = canvas.height-30;
24. **var** dx = 2;
25. **var** dy = -2;
27. /\* Las variables a continuación, tienen el siguiente significado:
29. Se define una paleta en la que rebotará la bola
30. La paleta está situada en la base de la pantalla de juego
31. Dicha paleta será controlada por la flecha izquierda y
32. la flecha derecha del teclado (luego será controlador por el ratón)
34. alturaPaleta: define la altura de la paleta en pixeles
35. anchuraPaleta: define la anchura de la paleta
37. NOTA: Estos dos valores determinan el tamaño de la paleta
38. La paleta se encuentra situada en la base de la pantalla
39. Para calcular la posición en X de la paleta, se debe tomar
40. el ancho del CANVAS, restarle la anchura de la paleta, y
41. el espacio que sobre debe dividirse entre dos
42. Esto garantiza que originalmente la paleta estará centrada
43. en la base de la pantalla
45. Al inicio del juego, aún no se ha presionado ninguna de las
46. flechas. Esta es la razón por la cual se definen dos variables que
47. "recuerdan" cual de las flechas se ha presionado, pero que
48. inicialmente están puestas a: false, indicando el estado inicial
49. Cuando se pulse cualquiera de las dos flechas, su valor será:
50. true (verdadero), y este valor permitira establecer en qué
51. dirección se debe mover la paleta (dentro del ciclo del juego)
52. Las variables son:
54. flechaDerechaPulsada
55. flechaIzquierdaPulsada
57. NOTA: Desde ahora debe tomarse en cuenta que cuando se pulse
58. cualquiera de las dos flechas, solamente se hará un
59. desplazamiento de la paleta a la izquierda o hacia la
60. derecha. Si se mantiene pulsada la tecla, la paleta se
61. continuará desplazando, hasta alcanzar el extremo derecho
62. o izquierdo de la pantalla del juego
63. \*/
65. **var** alturaPaleta = 15; // <---------------
66. **var** anchuraPaleta = 100; // <---------------
67. **var** paletaPosX = (canvas.width - anchuraPaleta) / 2; // <------------------
69. **var** flechaDerechapulsada= **false**; // <-----------------
70. **var** flechaIzquierdaPulsada = **false**; // <-----------------
72. /\* La instruccion: addEventListener, se utiliza para crear un
73. mecanismo de respuesta ante eventos que se produzcan en el juego
75. addEventListener "agregar un mecanismo que deteca y recibe eventos"
77. addEventListener recibe tres parámetros:
79. 1) El evento que se va a detectar
80. 2) El nombre que le asignamos a la función que responde ante el evento
81. 3) Valor true o false que determina la reacción ante el evento
83. Los dos primeros parámetros son fáciles de entender. Pero el tercero
84. requiere de una explicación adicional:
86. Para entender el tercer parámetro, primero hemos de saber lo que es
87. el flujo de eventos.
89. Supongamos que tenemos este tres objetos en la página:
91. <body>
92. <div>
93. <button>HAZME CLIC</button>
94. </div>
95. </body>
97. El <body> contiene un <div>, y dentro de él esta un <button>
99. Cuando hacemos clic en el botón no sólo lo estamos haciendo sobre él,
100. sino sobre los elementos que lo contienen en el árbol de la página,
101. es decir, hemos hecho clic, además, sobre el elemento <body> y sobre
102. el elemento <div>. Sí sólo hay una función asignada a una escucha
103. para el botón, no hay mayor problema, pero si además hay una
104. eschucha para el body y otra para el div,
105. ¿cuál es el orden en que se deben lanzar las tres funciones?
107. Para contestar a esa pregunta existe un modelo de comportamiento,
108. el flujo de eventos. Según éste, cuando se hace clic sobre un
109. elemento, el evento se propaga en dos fases, una que es la
110. captura —el evento comienza en el nivel superior del documento
111. y recorre los elementos de padres a hijos— y la otra la burbuja
112. —el orden inverso, ascendiendo de hijos a padres—.
114. Así, el orden por defecto de lanzamiento de las funciones
115. de escucha, sería: primero la función de escuch de body,
116. luego la función de escucha de div, y por último la función
117. de escucha de button.
119. Una vez visto esto, podemos comprender el tercer parámetro de addEventListener, que lo que hace es permitirnos escoger el orden de propagación:
121. true: El orden de propagación para el ejemplo sería, por tanto,
122. body-div-button
124. false: La propagación seguiría el modelo burbuja.
125. Así, el orden sería button-div-body.
127. NOTA: como en nuestro ejemplo utilizamos "false", estamos
128. eaccionando primero ante el evento sobre las teclas,
129. posteriormente sobre los eventos asociados al CANVAS.
130. ste es el mecanismo más usual, pero se utilizará "true"
131. n las situaciones que lo requieran
132. \*/
133. document.addEventListener("keydown", manejadorTeclaPresionada, **false**); // <---------
134. document.addEventListener("keyup", manejadorTeclaLiberada, **false**); // <--------
136. // Función que maneja tecla presionada
137. **function** manejadorTeclaPresionada(e) { // <----------------
138. **if**(e.keyCode == 39) {
139. /\* e: Es el evento que se produce, en este caso
140. tecla presionada. La propiedad: keyCode permite
141. descubrir de qué tecla se trata. Si el código es 39,
142. se ha presionado la flecha derecha. En este caso
143. se coloca la variable: flechaDerechaPulsada a true
144. \*/
145. flechaDerechaPulsada= **true**;
146. }
147. **else** **if**(e.keyCode == 37) {
148. /\* e: Es el evento que se produce, en este caso
149. tecla presionada. La propiedad: keyCode permite
150. descubrir de qué tecla se trata. Si el código es 37,
151. se ha presionado la flecha izquierda. En este caso
152. se coloca la variable: flechaIquierdaPulsada a true
153. \*/
154. flechaIzquierdaPulsada = **true**;
155. }
156. }
158. // Función que maneja tecla liberada
159. **function** manejadorTeclaLiberada(e) { // <--------------------
160. **if**(e.keyCode == 39) {
161. /\* Si la tecla liberada es la 39, se ha dejado de
162. presionar la flecha derecha. En este caso, la variable
163. se pone en: false
164. \*/
165. flechaDerechaPulsada= **false**;
166. }
167. **else** **if**(e.keyCode == 37) { // <-----------------
168. /\* Si la tecla liberada es la 37, se ha dejado de
169. presionar la flecha izquierda. En este caso, la variable
170. se pone en: false
171. \*/
172. flechaIzquierdaPulsada= **false**;
173. }
174. }
176. // Dibuja la bola. Código explicado en anteriores programas
177. **function** dibujarBola() {
178. ctx.beginPath();
179. ctx.arc(x, y, radioBola, 0, Math.PI\*2);
180. ctx.fillStyle = "#ff0000";
181. ctx.fill();
182. ctx.closePath();
183. }
185. **function** dibujarPaleta() { // <--------------------
186. // Se inicia el dibujo de la paleta
187. ctx.beginPath();
188. /\* Se crea un rectángulo utilizando la posición en X
189. El valor de Y está en la base de la pantalla menos la
190. altura de la paleta
191. Y a continuación se indica la anchura y la altura de la paleta
192. \*/
193. ctx.rect(paletaPosX, canvas.height-alturaPaleta, anchuraPaleta, alturaPaleta);
194. ctx.fillStyle = "#00ff00";
195. ctx.fill();
196. // Se "cierra" la paleta, terminando su dibujo
197. ctx.closePath();
198. }
200. // Función principal. A partir de aquí se origina el proceso
201. // general del juego
202. **function** dibujar() {
203. ctx.clearRect(0, 0, canvas.width, canvas.height);
205. // En primer lugar, dibuja la bola
206. dibujarBola();
208. // Seguidamente, dibuja la paleta
209. dibujarPaleta(); // <--------------------
211. /\* IMPORTANTE:
213. EL OPERADOR || es el operador lógico OR
214. Este operador se utiliza para indicar la condición de conjunción
215. SI SE CUMPLE UNA CONDICIÓN, O SE CUMPLE OTRA CONDICIÓN, ENTONCES
216. SE CUMPLE LA CONDICIÓN
218. EL OPERADOR && es el oerador lógico AND
219. Este operador se utiliza para indicar la condicion de disyunción
220. SI SE CUMPLE UNA CONDICIÓN, Y SE CUMPLE OTRA CONDICIÓN (simultánea), ENTONCES
221. SE CUMPLE LA CONDICIÓN
223. \*/
225. // Aquí se controla los límites a los que puede llegar la bola
226. // En caso de intentar sobrepasar dichos límites, se cambia
227. // el sentido del movimiento
228. // Este código se explicó en el anterior programa
229. **if**(x + dx > canvas.width-radioBola || x + dx < radioBola) { // <--------------
230. dx = -dx;
231. }
232. **if**(y + dy > canvas.height-radioBola || y + dy < radioBola) { // <--------------
233. dy = -dy;
234. }
236. /\* Si se ha pulsado la flecha derecha, y la paleta aún puede
237. desplazarse hacia la derecha sin que se sobrepase el límite de la
238. pantalla, entonces se procede a cambiar su posición
239. En este caso, la función: dibujarPaleta (la cual se ejecuta de
240. manera cíclica) redibujará la paleta en la nueva posición
241. \*/
242. **if**(flechaDerechaPulsada && paletaPosX < canvas.width-anchuraPaleta) { // <--------
243. // Se desplaza la paleta hacia la derecha
244. // Aquí, paletaPosX += 7 equivale a: paletaPosX = paletaPosX + 7
245. paletaPosX += 7;
246. }
247. **else** **if**(flechaIzquierdaPulsada && paletaPosX > 0) { // <------------
248. // Se desplaza la paleta hacia la izquierda
249. // Aquí, paletaPosX -= 7 equivale a: paletaPosX = paletaPosX - 7
250. paletaPosX -= 7;
251. }
253. x += dx;
254. y += dy;
255. }
257. /\* Con esta instrucción se crea un ciclo. Cada 10 milisegundos se
258. ejecuta la funcion: dibujar(). Esto genera el ciclo que permitirá
259. actualizar el juego, detectar eventos y cambiar el estado
260. de los objetos según las nuevas posiciones que ocupen los
261. elementos del juego
263. NOTA: La función que se ejecuta es: dibujar
264. Por tanto, dicha función es la encargada de "lanzar" el juego
265. y dentro de ella se realizarán las acciones que desencadenan
266. el juego como tal
267. \*/
268. setInterval(dibujar, 10);
269. </script>
271. </body>
272. </html>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 4. Interfaz fase 3**

En la figura 4 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.

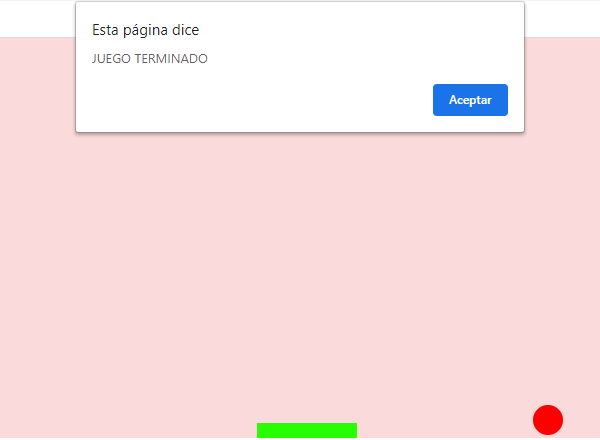
# FASE 4: Fin del juego

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 código: (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: #FADADA; display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
10. <canvas id="miCanvas" width="600" height="400"></canvas>
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. *\*/*
21. **var** canvas = document.getElementById("miCanvas");
22. **var** ctx = canvas.getContext("2d");
24. **var** radioBola = 15;
25. **var** x = canvas.width/2;
26. **var** y = canvas.height-30;
27. **var** dx = 2;
28. **var** dy = -2;
30. **var** alturaPaleta = 15;
31. **var** anchuraPaleta = 100;
32. **var** paletaPosX = (canvas.width-anchuraPaleta)/2;
34. **var** flechaDerechaPresionada = **false**;
35. **var** flechaIzquierdaPresionada = **false**;
37. document.addEventListener("keydown", manejadorTeclaPresionada, **false**);
38. document.addEventListener("keyup", manejadorTeclaLiberada, **false**);
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. }
57. **function** dibujarBola() {
58. ctx.beginPath();
59. ctx.arc(x, y, radioBola, 0, Math.PI\*2);
60. ctx.fillStyle = "#FF0000";
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 = "#27FF00";
68. ctx.fill();
69. ctx.closePath();
70. }
72. **function** dibujar() {
73. ctx.clearRect(0, 0, canvas.width, canvas.height);
75. dibujarBola();
76. dibujarPaleta();
78. **if**(x + dx > canvas.width-radioBola || x + dx < radioBola) {
79. dx = -dx;
80. }
81. **if**(y + dy < radioBola) {
82. dy = -dy;
83. }
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("JUEGO TERMINADO");
107. document.location.reload();
108. }
109. }
111. **if**(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
112. paletaPosX += 7;
113. }
114. **else** **if**(flechaIzquierdaPresionada && paletaPosX > 0) {
115. paletaPosX -= 7;
116. }
118. x += dx;
119. y += dy;
120. }
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>
129. </body>
130. </html>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 5. Interfaz fase 4**

En la gráfica 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 “JUEGO TERMINADO” que por ende significa que el juego se ha acabado.

En el siguiente apartado se explicará la siguiente fase del juego.

# FASE 5: Muro de ladrillos

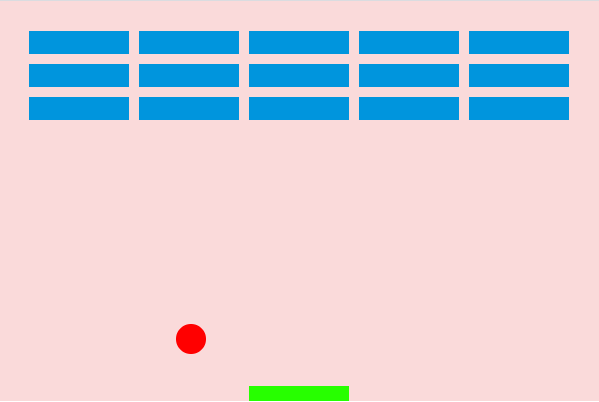
En esta parte del juego crearemos unas variables las cuales crearan un muro de ladrillos dentro del juego en los cuales rebotara la bola.

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.

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: #FADADA; display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
10. <canvas id="miCanvas" width="600" height="400"></canvas>
12. <script>
13. **var** canvas = document.getElementById("miCanvas");
14. **var** ctx = canvas.getContext("2d");
15. **var** radioBola = 15;
16. **var** x = canvas.width/2;
17. **var** y = canvas.height-30;
18. **var** dx = 2;
19. **var** dy = -2;
20. **var** alturaPaleta = 15;
21. **var** anchuraPaleta = 100;
22. **var** paletaPosX = (canvas.width-anchuraPaleta)/2;
23. **var** flechaDerechaPresionada = **false**;
24. **var** flechaIzquierdaPresionada = **false**;
26. */\* NUEVAS VARIABLES asociadas a los ladrillos*
27. *\*/*
28. **var** nroFilasLadrillos = 5;
29. **var** nroColumnasLadrillos = 3;
30. **var** anchoLadrillo = 100;
31. **var** alturaLadrillo = 23;
32. **var** rellenoLadrillo = 10;
33. **var** vacioSuperiorLadrillo = 30;
34. **var** vacioIzquierdoLadrillo = 30;
36. *// Crea el conjunto de ladrillos. Inicialmente, vacío*
37. **var** ladrillos = [];
39. *// Recorre cinco columnas*
40. **for**(**var** columna=0; columna<nroColumnasLadrillos; columna++) {
41. *// Define la primera columna. Es una lista vertical*
42. ladrillos[columna] = [];
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*

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:*
56. *x: Su coordenada horizontal EN LA PANTALLA*
57. *y: Su coordenada vertical EN LA PANTALLA*
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. }
67. document.addEventListener("keydown", manejadorTeclaPresionada, **false**);
68. document.addEventListener("keyup", manejadorTeclaLiberada, **false**);
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. }
87. **function** dibujarBola() {
88. ctx.beginPath();
89. ctx.arc(x, y, radioBola, 0, Math.PI\*2);
90. ctx.fillStyle = "#FF0000";
91. ctx.fill();
92. ctx.closePath();
93. }
95. **function** dibujarPaleta() {
96. ctx.beginPath();
97. ctx.rect(paletaPosX, canvas.height-alturaPaleta, anchuraPaleta, alturaPaleta);
98. ctx.fillStyle = "#27FF00";
99. ctx.fill();
100. ctx.closePath();
101. }
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))+vacioIzquierdoLadrillo;
117. *// Repite el proceso para calcular la coordenada y del ladrillo*
118. **var** brickY = (columna\*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLadrillo;
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;
124. *// IGUAL PARA EL VALOR y EN PANTALLA*
125. ladrillos[columna][fila].y = brickY;
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();
137. *// COMO SE RECORRE TODO EL CICLO, SE DIBUJAN TODOS LOS LADRILLOS*
138. }
139. }
140. }
142. **function** dibujar() {
143. ctx.clearRect(0, 0, canvas.width, canvas.height);
145. *// DIBUJA EL CONJUNTO DE LADRILLOS*
146. dibujarLadrillos();
148. dibujarBola();
149. dibujarPaleta();
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("JUEGO TERMINADO");
165. *// RECARGA LA PÁGINA - El juego vuelve a empezar*
166. document.location.reload();
167. }
168. }
170. **if**(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
171. paletaPosX += 7;
172. }
173. **else** **if**(flechaIzquierdaPresionada && paletaPosX > 0) {
174. paletaPosX -= 7;
175. }
177. x += dx;
178. y += dy;
179. }
181. **var** juego = setInterval(dibujar, 10);
182. </script>
184. </body>
185. </html>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 6. Interfaz fase 5**

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.

# FASE 6: Detección de colisiones

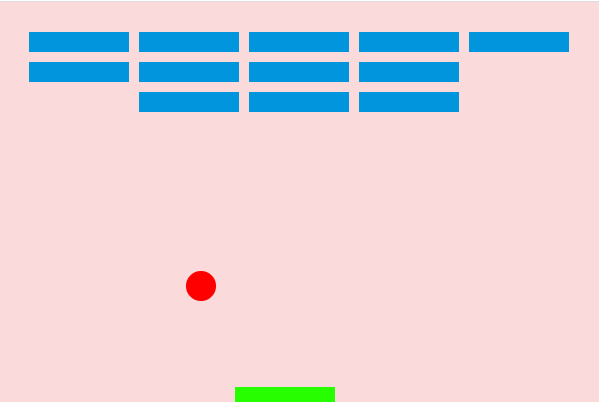
En esta parte del programa realizaremos la función que hará que se detecte la colisión de la bola con alguno de los ladrillos, al ocurrir esto el ladrillo desaparecerá.

Esta es llamada: function **deteccionColision**(), la cual se le agrega un variable temporal que analiza la fila y columna en el cual el ladrillo es impactado, para que este desaparezca.

También se crean las siguientes variables: la primera es clearInterval(juego); la cual hace que se detenga el ciclo del juego, otra es alert("JUEGO TERMINADO"); la cual hace que al perder el juego salga un letrero con la palabra JUEGO TERMINADO y por último la siguiente variable document.location.reload(); que hace que el juego se recargue nuevamente y pueda empezar otra vez.

1. <!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: #FADADA; display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
10. <canvas id="miCanvas" width="600" height="400"></canvas>
12. <script>
13. **var** canvas = document.getElementById("miCanvas");
14. **var** ctx = canvas.getContext("2d");
16. **var** radioBola = 15;
17. **var** x = canvas.width/2;
18. **var** y = canvas.height-30;
19. **var** dx = 2;
20. **var** dy = -2;
22. **var** alturaPaleta = 15;
23. **var** anchuraPaleta = 100;
24. **var** paletaPosX = (canvas.width-anchuraPaleta)/2;
26. **var** flechaDerechaPresionada = **false**;
27. **var** flechaIzquierdaPresionada = **false**;
29. **var** nroFilasLadrillos = 5;
30. **var** nroColumnasLadrillos = 3;
31. **var** anchuraLadrillo = 100;
32. **var** alturaLadrillo = 20;
33. **var** rellenoLadrillo = 10;
34. **var** vacioSuperiorLadrillo = 30;
35. **var** vacioIzquierdoLadrillo = 30;
37. **var** ladrillos = [];
38. **for**(**var** c=0; c<nroColumnasLadrillos; c++) {
39. ladrillos[c] = [];
40. **for**(**var** f=0; f<nroFilasLadrillos; f++) {
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:*
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*
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. }
63. document.addEventListener("keydown", manejadorTeclaPresionada, **false**);
64. document.addEventListener("keyup", manejadorTeclaLiberada, **false**);
66. **function** manejadorTeclaPresionada(e) {
67. **if**(e.keyCode == 39) {
68. flechaDerechaPresionada = **true**;
69. }
70. **else** **if**(e.keyCode == 37) {
71. flechaIzquierdaPresionada = **true**;
72. }
73. }
75. **function** manejadorTeclaLiberada(e) {
76. **if**(e.keyCode == 39) {
77. flechaDerechaPresionada = **false**;
78. }
79. **else** **if**(e.keyCode == 37) {
80. flechaIzquierdaPresionada = **false**;
81. }
82. }
84. *// EN ESTA FUNCIÓN SE DETECTA LA COLISIÓN DE LA BOLA CON EL LADRILLO*
86. **function** deteccionColision() {
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++) {
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];
96. *// SI EL LADRILLO ES VISIBLE, se debe verificar si entra en contacto con la bola*
97. **if**(b.status == 1) {
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. *\*/*
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. }
116. **function** dibujarBola() {
117. ctx.beginPath();
118. ctx.arc(x, y, radioBola, 0, Math.PI\*2);
119. ctx.fillStyle = "#FF0000";
120. ctx.fill();
121. ctx.closePath();
122. }
124. **function** dibujarPaleta() {
125. ctx.beginPath();
126. ctx.rect(paletaPosX, canvas.height-alturaPaleta, anchuraPaleta, alturaPaleta);
127. ctx.fillStyle = "#27FF00";
128. ctx.fill();
129. ctx.closePath();
130. }
132. **function** dibujarLadrillos() {
133. **for**(**var** c=0; c<nroColumnasLadrillos; c++) {
134. **for**(**var** f=0; f<nroFilasLadrillos; f++) {
136. */\* IMPORTANTE:*
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*
142. *\*/*
144. **if**(ladrillos[c][f].status == 1) {
146. *// SE DIBUJA EL LADRILLO*
147. **var** brickX = (f\*(anchuraLadrillo+rellenoLadrillo))+vacioIzquierdoLadrillo;
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. }
161. **function** dibujar() {
162. ctx.clearRect(0, 0, canvas.width, canvas.height);
163. dibujarLadrillos();
164. dibujarBola();
165. dibujarPaleta();
166. deteccionColision();
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("JUEGO TERMINADO");
183. *// Recarga la página, para iniciar de nuevo el juego*
184. document.location.reload();
185. }
186. }
188. **if**(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
189. paletaPosX += 7;
190. }
191. **else** **if**(flechaIzquierdaPresionada && paletaPosX > 0) {
192. paletaPosX -= 7;
193. }
195. x += dx;
196. y += dy;
197. }
199. **var** juego = setInterval(dibujar, 10);
200. </script>
202. </body>
203. </html>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 7. Interfaz fase 6**

En la figura 7 podemos observar cómo algunos ladrillos se desaparecieron luego de ser golpeados por la bola.

En el siguiente apartado se explicará la siguiente fase del juego.

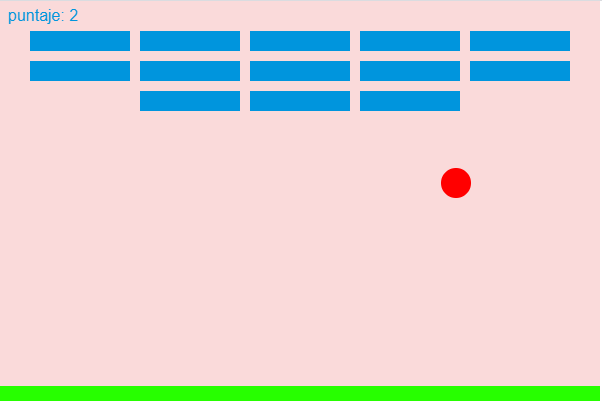
# FASE 7: Contar puntos y ganar

En esta fase se crea una variable que calcula puntos analizando si un ladrillo fue golpeado, los suma hasta que se gana el juego.

Esta variable es 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: #FADADA; display: block; margin: 0 auto; }</style>
10. </head>
11. <body>
13. <canvas id="miCanvas" width="600" height="400"></canvas>
15. <script>
16. **var** canvas = document.getElementById("miCanvas");
17. **var** ctx = canvas.getContext("2d");
19. **var** radioBola = 15;
20. **var** x = canvas.width/2;
21. **var** y = canvas.height-30;
22. **var** dx = 2;
23. **var** dy = -2;
24. **var** alturaPaleta = 15;
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 = 600;
30. **var** paletaPosX = (canvas.width-anchuraPaleta)/2;
31. **var** flechaDerechaPresionada = **false**;
32. **var** flechaIzquierdaPresionada = **false**;
34. **var** nroFilasLadrillos = 5;
35. **var** nroColumnasLadrillos = 3;
36. **var** anchuraLadrillos = 100;
37. **var** alturaLadrillos = 20;
38. **var** rellenoLadrillos = 10;
39. **var** vacioSuperiorLadrillo = 30;
40. **var** vacioIzquierdoLadrillo = 30;
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;
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. }
55. document.addEventListener("keydown", manejadorTeclaPresionada, **false**);
56. document.addEventListener("keyup", manejadorTeclaLiberada, **false**);
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 && y > b.y && y < b.y+alturaLadrillos) {
80. dy = -dy;
81. b.estado = 0;
83. *// LA INSTRUCCIÓN puntaje++ EQUIVALE A: puntaje = puntaje + 1*
84. *// ----------------------------------------------------------*
85. *// EN ESTE PUNTO DEL CÓDIGO LA BOLA HA IMPACTADO UN LADRILLO*
86. *// POR ESTE MOTIVO, SE INCREMENTA EL VALOR DE puntaje*
87. *// Si el puntaje es igual al número total de ladrillos (valor que*
88. *// se obtiene multiplicando el número de filas de ladrillos por el*
89. *// número de columnas de ladrillos), entonces el jugador ha ganado*
90. puntaje++;
91. **if**(puntaje == nroFilasLadrillos\*nroColumnasLadrillos) {
92. alert("USTED GANA! FELICITACIONES!!!");
93. document.location.reload();
94. }
96. }
97. }
98. }
99. }
100. }
102. **function** dibujarBola() {
103. ctx.beginPath();
104. ctx.arc(x, y, radioBola, 0, Math.PI\*2);
105. ctx.fillStyle = "#FF0000";
106. ctx.fill();
107. ctx.closePath();
108. }
110. **function** dibujarPaleta() {
111. ctx.beginPath();
112. ctx.rect(paletaPosX, canvas.height-alturaPaleta, anchuraPaleta, alturaPaleta);
113. ctx.fillStyle = "#27FF00";
114. ctx.fill();
115. ctx.closePath();
116. }
118. **function** dibujarLadrillos() {
119. **for**(**var** c=0; c<nroColumnasLadrillos; c++) {
120. **for**(**var** r=0; r<nroFilasLadrillos; r++) {
121. **if**(ladrillos[c][r].estado == 1) {
122. **var** posXLadrillo = (r\*(anchuraLadrillos+rellenoLadrillos))+vacioIzquierdoLadrillo;
123. **var** posYLadrillo = (c\*(alturaLadrillos+rellenoLadrillos))+vacioSuperiorLadrillo;
124. ladrillos[c][r].x = posXLadrillo;
125. ladrillos[c][r].y = posYLadrillo;
126. ctx.beginPath();
127. ctx.rect(posXLadrillo, posYLadrillo, anchuraLadrillos, alturaLadrillos);
128. ctx.fillStyle = "#0095DD";
129. ctx.fill();
130. ctx.closePath();
131. }
132. }
133. }
134. }
136. **function** dibujarPuntaje() {
137. ctx.font = "16px Arial";
138. ctx.fillStyle = "#0095DD";
139. ctx.fillText("puntaje: "+puntaje, 8, 20);
140. }
142. **function** dibujar() {
143. ctx.clearRect(0, 0, canvas.width, canvas.height);
144. dibujarLadrillos();
145. dibujarBola();
146. dibujarPaleta();
147. dibujarPuntaje();
148. detectarColision();
150. **if**(x + dx > canvas.width-radioBola || x + dx < radioBola) {
151. dx = -dx;
152. }
153. **if**(y + dy < radioBola) {
154. dy = -dy;
155. }
156. **else** **if**(y + dy > canvas.height-radioBola) {
157. **if**(x > paletaPosX && x < paletaPosX + anchuraPaleta) {
158. dy = -dy;
159. }
160. **else** {
161. clearInterval(juego);
162. alert("JUEGO TERMINADO");
163. document.location.reload();
164. }
165. }
167. **if**(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
168. paletaPosX += 7;
169. }
170. **else** **if**(flechaIzquierdaPresionada && paletaPosX > 0) {
171. paletaPosX -= 7;
172. }
174. x += dx;
175. y += dy;
176. }
178. **var** juego = setInterval(dibujar, 10);
179. </script>
181. </body>
182. </html>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 8. Interfaz fase 7**

En la figura 8 se puede observar algunos ladrillos que han desaparecido y en la parte superior izquierda esta el conteo de los puntos.

En el siguiente apartado se explicará la siguiente fase del juego.

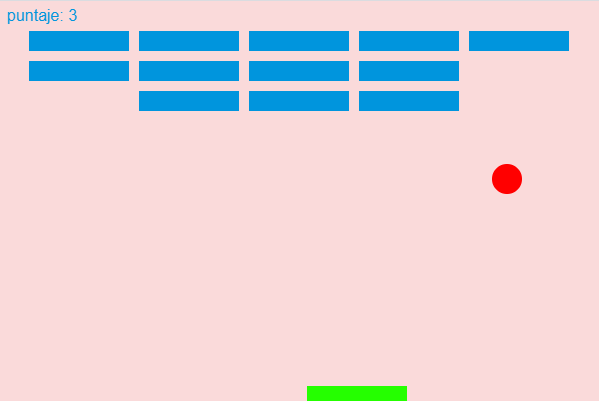
# FASE 8: Controlando el ratón

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

Esto se hace mediante una función llamada function manejadorRaton(e) a la cual se le da una variable “e” 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: #FADADA; display: block; margin: 0 auto; }</style>
7. </head>
8. <body>
10. <canvas id="miCanvas" width="600" height="400"></canvas>
12. <script>
13. **var** canvas = document.getElementById("miCanvas");
14. **var** ctx = canvas.getContext("2d");
16. **var** radioBola = 15;
17. **var** x = canvas.width/2;
18. **var** y = canvas.height-30;
19. **var** dx = 2;
20. **var** dy = -2;
22. **var** alturaPaleta = 15;
23. **var** anchuraPaleta = 100;
24. **var** paletaPosX = (canvas.width-anchuraPaleta)/2;
26. **var** flechaDerechaPresionada = **false**;
27. **var** flechaIzquierdaPresionada = **false**;
29. **var** nroFilasLadrillos = 5;
30. **var** nroColumnasLadrillos = 3;
31. **var** anchuraLadrillo = 100;
32. **var** alturaLadrillo = 20;
33. **var** rellenoLadrillo = 10;
34. **var** vacioSuperiorLadrillo = 30;
35. **var** vacioIzquierdoLadrillo = 30;
37. **var** puntaje = 0;
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. }
47. document.addEventListener("keydown", manejadorTeclaPresionada, **false**);
48. document.addEventListener("keyup", manejadorTeclaLiberada, **false**);
50. *// PARA DETECTAR EL MOVIMIENTO DEL RATÓN, SE COLOCA UN ESCUCHADOR (listener)*
51. *// AL EVENTO "mousemove"*
52. document.addEventListener("mousemove", manejadorRaton, **false**);
54. **function** manejadorTeclaPresionada(e) {
55. **if**(e.keyCode == 39) {
56. flechaDerechaPresionada = **true**;
57. }
58. **else** **if**(e.keyCode == 37) {
59. flechaIzquierdaPresionada = **true**;
60. }
61. }
63. **function** manejadorTeclaLiberada(e) {
64. **if**(e.keyCode == 39) {
65. flechaDerechaPresionada = **false**;
66. }
67. **else** **if**(e.keyCode == 37) {
68. flechaIzquierdaPresionada = **false**;
69. }
70. }
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. }
123. **function** dibujarBola() {
124. ctx.beginPath();
125. ctx.arc(x, y, radioBola, 0, Math.PI\*2);
126. ctx.fillStyle = "#FF0000";
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 = "#25FF00";
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))+vacioIzquierdoLadrillo;
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. }
160. **function** dibujar() {
161. ctx.clearRect(0, 0, canvas.width, canvas.height);
162. dibujarLadrillos();
163. dibujarBola();
164. dibujarPaleta();
165. dibujarPuntaje();
166. detectarColision();
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("JUEGO TERMINADO");
181. document.location.reload();
182. }
183. }
185. **if**(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
186. paletaPosX += 7;
187. }
188. **else** **if**(flechaIzquierdaPresionada && paletaPosX > 0) {
189. paletaPosX -= 7;
190. }
192. x += dx;
193. y += dy;
194. }
196. **var** juego = setInterval(dibujar, 10);
197. </script>
199. </body>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 9. Interfaz fase 8**

En el siguiente apartado se explicará la fase final del juego.

# FASE 9: Finalizando el juego

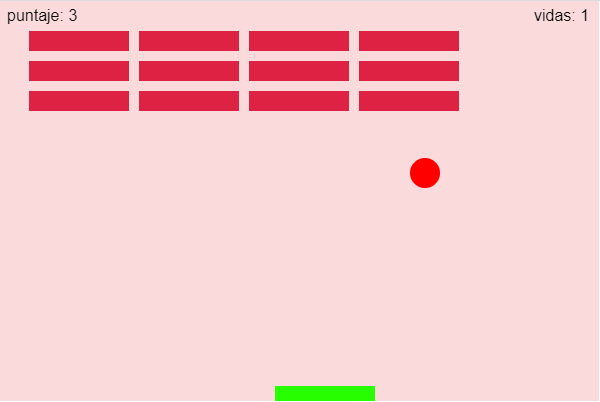
En esta parte del programa ya se agregan los toques finales tales como vidas para el jugador, ocultar el mouse para que no estorbe la visión etc.

Se hace una variable var vidas = 3 para 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: #FADADA; display: block; margin: 0 auto; \* {cursor: none;} } </style>
10. </head>
11. <body>
13. <canvas id="miCanvas" width="600" height="400"></canvas>
15. <script>
16. **var** canvas = document.getElementById("miCanvas");
17. **var** ctx = canvas.getContext("2d");
19. **var** bolaRadio = 15;
20. **var** x = canvas.width/2;
21. **var** y = canvas.height-30;
22. **var** dx = 2;
23. **var** dy = -2;
25. **var** alturaPaleta = 15;
26. **var** anchuraPaleta = 100;
27. **var** paletaPosX = (canvas.width-anchuraPaleta)/2;
29. **var** flechaDerechaPresionada = **false**;
30. **var** flechaIzquierdaPresionada = **false**;
32. **var** nroFilasLadrillos = 5;
33. **var** nroColumnasLadrillos = 3;
34. **var** anchuraLadrillo = 100;
35. **var** alturaLadrillo = 20;
36. **var** rellenoLadrillo = 10;
37. **var** vacioSuperiorLadrillo = 30;
38. **var** vacioIzquierdoLadrillo = 30;
40. **var** puntaje = 0;
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;

48. *// ESTA VARIABLE DEFINE UN COLOR*
49. *// Se pueden utilizar otros colores para los diferentes elementos del juego*
50. **var** colorFigura = "#ff0000";
51. **var** colorBola = "#FF0000";
52. **var** colorPaleta = "#27FF00";
53. **var** colorLadrillo = "#dd2244";
54. **var** colorTexto = "#000000";
56. *// ESTA INSTRUCCIÓN OCULTA EL CURSOR DEL RATON (DENTRO DEL CANVAS)*
57. canvas.style.cursor = 'none';
59. **var** ladrillos = [];
60. **for**(**var** c=0; c<nroColumnasLadrillos; c++) {
61. ladrillos[c] = [];
62. **for**(**var** f=0; f<nroFilasLadrillos; f++) {
63. ladrillos[c][f] = { x: 0, y: 0, estado: 1 };
64. }
65. }
67. document.addEventListener("keydown", manejadorTeclaPresionada, **false**);
68. document.addEventListener("keyup", manejadorTeclaLiberada, **false**);
69. document.addEventListener("mousemove", manejadorRaton, **false**);
71. **function** manejadorTeclaPresionada(e) {
72. **if**(e.keyCode == 39) {
73. flechaDerechaPresionada = **true**;
74. }
75. **else** **if**(e.keyCode == 37) {
76. flechaIzquierdaPresionada = **true**;
77. }
78. }
80. **function** manejadorTeclaLiberada(e) {
81. **if**(e.keyCode == 39) {
82. flechaDerechaPresionada = **false**;
83. }
84. **else** **if**(e.keyCode == 37) {
85. flechaIzquierdaPresionada = **false**;
86. }
87. }
89. **function** manejadorRaton(e) {
90. **var** posXRatonDentroDeCanvas = e.clientX - canvas.offsetLeft;
91. **if**(posXRatonDentroDeCanvas > 0 && posXRatonDentroDeCanvas < canvas.width) {
92. paletaPosX = posXRatonDentroDeCanvas - anchuraPaleta/2;
93. }
94. }
96. **function** detectarColision() {
97. **for**(**var** c=0; c<nroColumnasLadrillos; c++) {
98. **for**(**var** f=0; f<nroFilasLadrillos; f++) {
99. **var** b = ladrillos[c][f];
100. **if**(b.estado == 1) {
101. **if**(x > b.x && x < b.x+anchuraLadrillo && y > b.y && y < b.y+alturaLadrillo) {
102. dy = -dy;
103. b.estado = 0;
104. puntaje++;
105. **if**(puntaje == nroFilasLadrillos\*nroColumnasLadrillos) {
106. alert("USTED GANA, FELICITACIONES!");
107. document.location.reload();
108. }
109. }
110. }
111. }
112. }
113. }
115. **function** dibujarBola() {
116. ctx.beginPath();
117. ctx.arc(x, y, bolaRadio, 0, Math.PI\*2);
118. *// SE UTILIZA EL COLOR PREVIAMENTE DEFINIDO*
119. ctx.fillStyle = colorBola;
120. ctx.fill();
121. ctx.closePath();
122. }
123. **function** dibujarPaleta() {
124. ctx.beginPath();
125. ctx.rect(paletaPosX, canvas.height-alturaPaleta, anchuraPaleta, alturaPaleta);
126. ctx.fillStyle = colorPaleta;
127. ctx.fill();
128. ctx.closePath();
129. }
130. **function** dibujarLadrillos() {
131. **for**(**var** c=0; c<nroColumnasLadrillos; c++) {
132. **for**(**var** f=0; f<nroFilasLadrillos; f++) {
133. **if**(ladrillos[c][f].estado == 1) {
134. **var** ladrilloX = (f\*(anchuraLadrillo+rellenoLadrillo))+vacioIzquierdoLadrillo;
135. **var** ladrilloY = (c\*(alturaLadrillo+rellenoLadrillo))+vacioSuperiorLadrillo;
136. ladrillos[c][f].x = ladrilloX;
137. ladrillos[c][f].y = ladrilloY;
138. ctx.beginPath();
139. ctx.rect(ladrilloX, ladrilloY, anchuraLadrillo, alturaLadrillo);
140. ctx.fillStyle = colorLadrillo;
141. ctx.fill();
142. ctx.closePath();
143. }
144. }
145. }
146. }
148. **function** dibujarPuntaje() {
149. ctx.font = "16px Arial";
150. ctx.fillStyle = colorTexto;
151. ctx.fillText("puntaje: "+puntaje, 8, 20);
152. }
154. **function** dibujarVidas() {
155. ctx.font = "16px Arial";
156. ctx.fillStyle = colorTexto;
157. *// SE MUESTRA EL NÚMERO DE VIDAS DISPONIBLES*
158. ctx.fillText("vidas: "+vidas, canvas.width-65, 20);
159. }
161. **function** dibujar() {
162. ctx.clearRect(0, 0, canvas.width, canvas.height);
163. dibujarLadrillos();
164. dibujarBola();
165. dibujarPaleta();
166. dibujarPuntaje();
167. dibujarVidas();
168. detectarColision();
170. **if**(x + dx > canvas.width-bolaRadio || x + dx < bolaRadio) {
171. dx = -dx;
172. }
173. **if**(y + dy < bolaRadio) {
174. dy = -dy;
175. }
176. **else** **if**(y + dy > canvas.height-bolaRadio) {
177. **if**(x > paletaPosX && x < paletaPosX + anchuraPaleta) {
178. dy = -dy;
179. }
180. **else** {
181. *// SI SE PRODUCE UN CONTACTO DE LA BOLA CON LA BASE DEL CANVAS*
182. *// SE PIERDE UNA VIDA. PARA ELLO, LA INSTRUCCIÓN vidas--;*
183. *// LO CUAL EQUIVALE A: vidas = vidas - 1*
184. vidas--;
185. **if**(!vidas) {
186. *// SI vidas == 0 (lo cual también puede escribir: !vidas)*
187. *// EL JUGADOR HA PERDIDO*
188. alert("JUEGO TERMINADO");
189. document.location.reload();
190. }
191. **else** {
192. *// SI vidas > 0 (diferente de CERO) EL JUEGO CONTINUA*
193. x = canvas.width/2;
194. y = canvas.height-30;
195. dx = 3;
196. dy = -3;
197. paletaPosX = (canvas.width-anchuraPaleta)/2;
198. }
199. }
200. }
202. **if**(flechaDerechaPresionada && paletaPosX < canvas.width-anchuraPaleta) {
203. paletaPosX += 7;
204. }
205. **else** **if**(flechaIzquierdaPresionada && paletaPosX > 0) {
206. paletaPosX -= 7;
207. }
209. x += dx;
210. y += dy;
212. *// ESTE ES UN SEGUNDO MÉTODO PARA REALIZAR LA ANIMACIÓN DEL JUEGO*
213. *// LA INSTRUCCIÓN: requestAnimationFrame SE EJECUTA 60 VECES POR SEGUNDO*
214. *// Y AL EJECUTARSE LLAMA A LA FUNCIÓN ENTRE PARÉNTESIS*
215. *// POR TANTO, dibujar SE EJECUTA 60 VECES POR SEGUNDO*
216. *// GENERANDO EL CICLO DEL JUEGO*
217. requestAnimationFrame(dibujar);
218. }
220. dibujar();
221. </script>
223. </body>
224. </html>

Al ejecutar este código se obtiene la siguiente interfaz visual:



**Grafica 10. Interfaz del juego terminado**

En la figura 10 podemos ver el juego terminado con todos sus detalles, el número de vidas, el puntaje etc. listo para ser jugado.

# CONCLUSIONES

En conclusión, se puede ver que experimentando con cada fase yendo paso por paso y buscando material de ayuda por internet se puede hacer un juego con HTML5 y JavaScript para emplearlo a al ámbito web.

Esto nos ayuda a aprender y tener bases sobre programación no solo para desarrollar páginas web, sino también para aplicarla a otros campos, Esto va a ser importante para el mundo tecnológico en el que estamos viviendo.

# BIBLIOGRAFÍA

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