

# CONTENIDO

[1 CONTENIDO 1](#_30j0zll)

[2 PRESENTACIÓN 2](#_1fob9te)

[3 FASE 1: Dibujar y mover una bola 3](#_3znysh7)

[4 FASE 2: Rebotando en las paredes 6](#_2et92p0)

[5 FASE 3: Control de la pala y el teclado 9](#_3dy6vkm)

[6 FASE 4: Fin del juego 18](#_1t3h5sf)

[7 FASE 5: Muro de ladrillos 22](#_4d34og8)

[8 FASE 6: Detección de colisiones 28](#_2s8eyo1)

[9 FASE 7: Contar puntos y ganar 33](#_17dp8vu)

[10 FASE 8: Controlando el ratón 39](#_3rdcrjn)

[11 FASE 9: Finalizando el juego 45](#_26in1rg)

[12 CONCLUSIONES 51](#_lnxbz9)

[13 BIBLIOGRAFÍA 52](#_35nkun2)

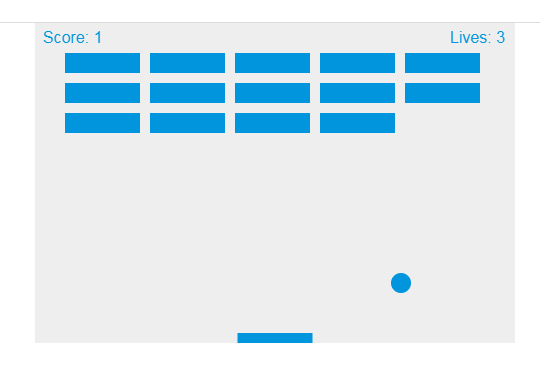
# PRESENTACIÓN

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: ALEXANDER GONZALEZ. JUAN ESTEBAN**

# FASE 1: Dibujar y mover una bola

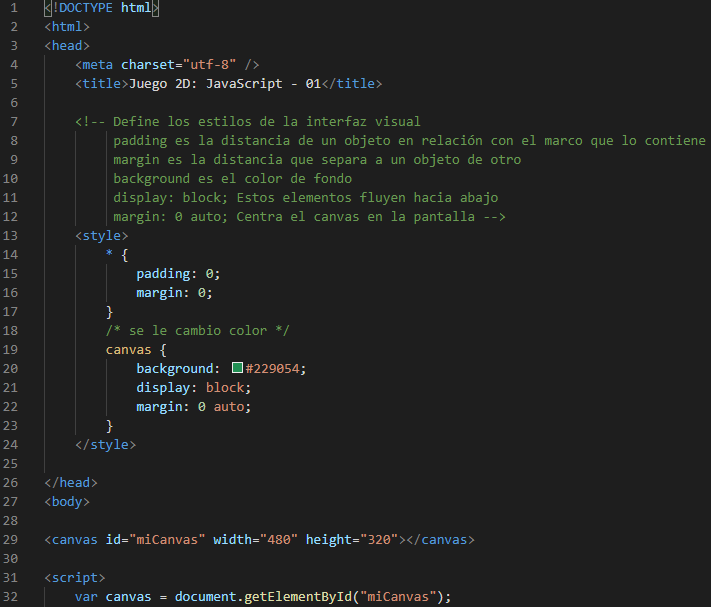
El primer paso consiste en elaborar una página HTML básica. Agregaremos a dicha página un elemento CANVAS, el cual nos servirá como base para el desarrollo del juego 2D.

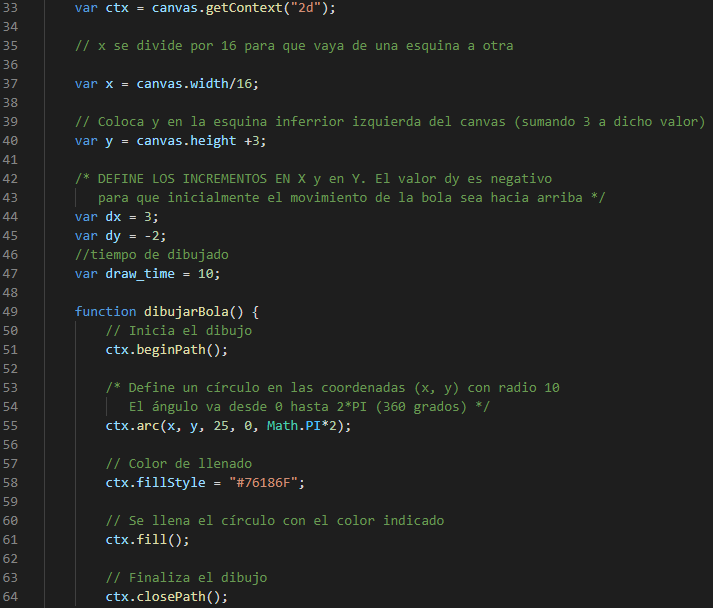
El código JavaScript que operará sobre el CANVAS debe encerrarse entre las etiquetas <script>…</script>

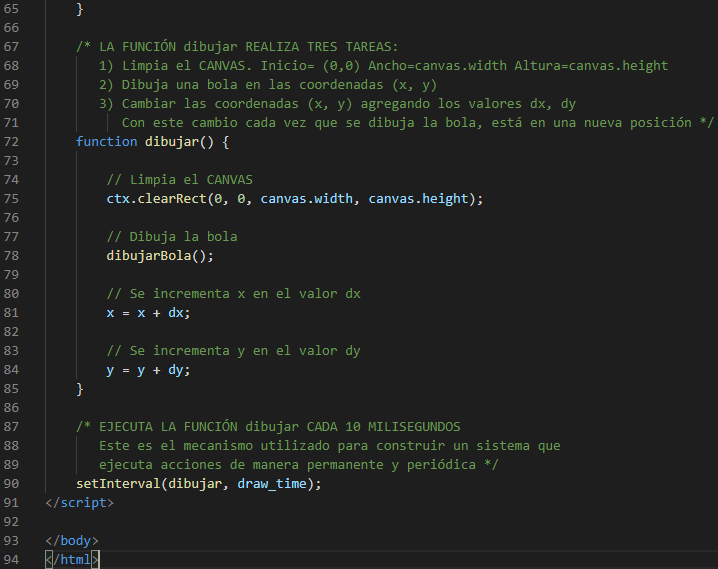
La correcta visualización del CANVAS requiere de la adición de algunas características de estilo. Una vez hecho esto, se procede a establecer la codificación pertinente del JavaScript. Debe notarse la inclusión de algunas variables que definen la funcionalidad del juego en sus aspectos básicos: las coordenadas 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 cambia los valores de las coordenadas. Finalmente, la función setInterval(dibujar, 10), llama a la función dibujar cada 10 milisegundos.

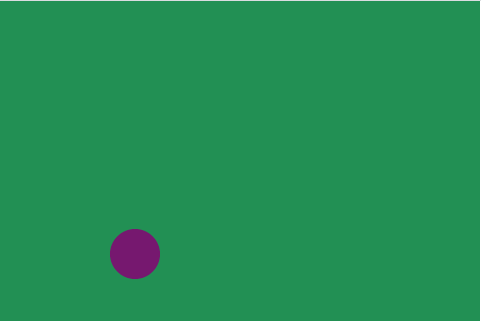
El código fuente del programa es el siguiente (para darle formato, se deben seguir las instrucciones disponibles en: <https://trabajonomada.com/insertar-codigo-word/> y seguidamente utilizar el enlace: <http://qbnz.com/highlighter/php_highlighter.php>)







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.

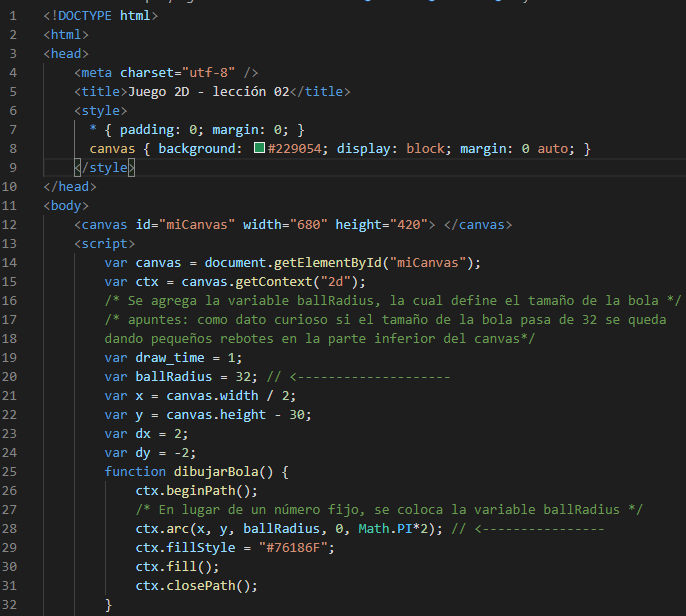
# FASE 2: Rebotando en las paredes

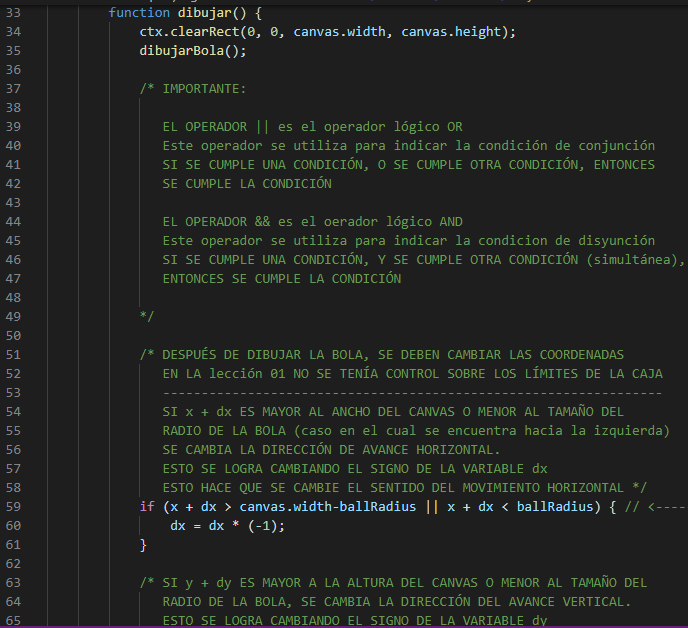
El segundo paso consiste en elaborar los límites permitidos a los que la bola puede llegar y en los que rebotara y así mantenerse dentro del cuadro asignado.

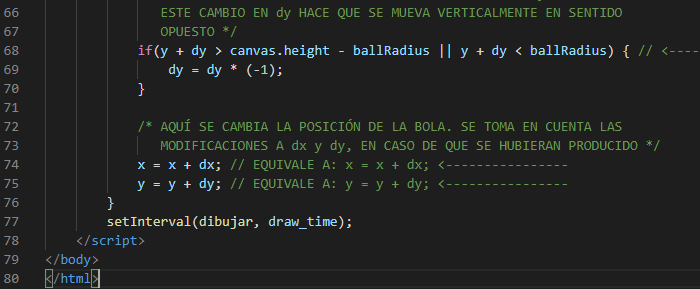
En este paso se crean dos condiciones las cuales generan los limites permitidos a los que la bola puede llegar y va a rebotar:

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.







Al ejecutar este código se obtiene la siguiente interfaz visual:



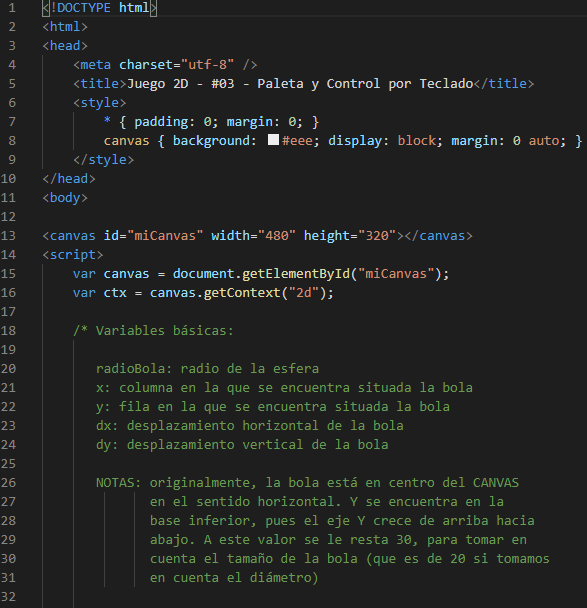
En la figura 3 podemos observar a la bola rebotando y cumpliendo con los límites anteriormente definidos en las condiciones. Como dato curioso si el radio de la bola es mayor a 32 empieza a dar pequeños rebotes en la parte inferior de canvas

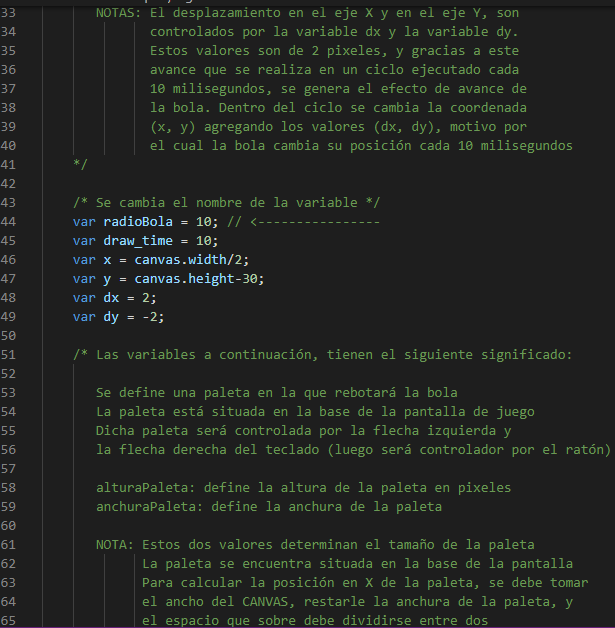
En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

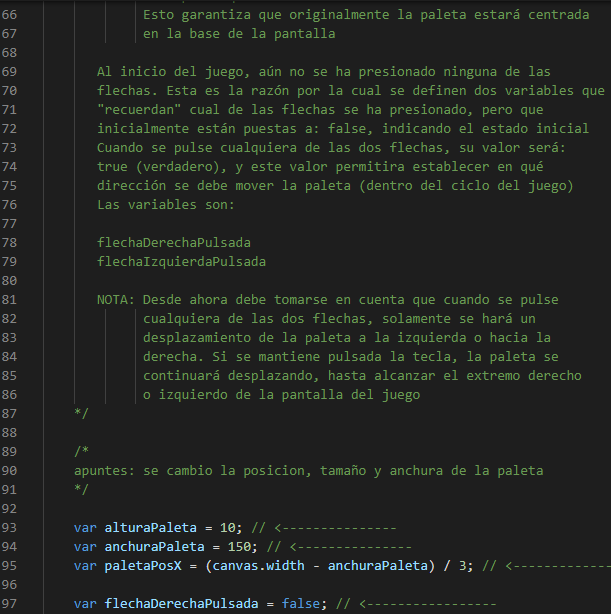
# FASE 3: Control de la pala y el teclado

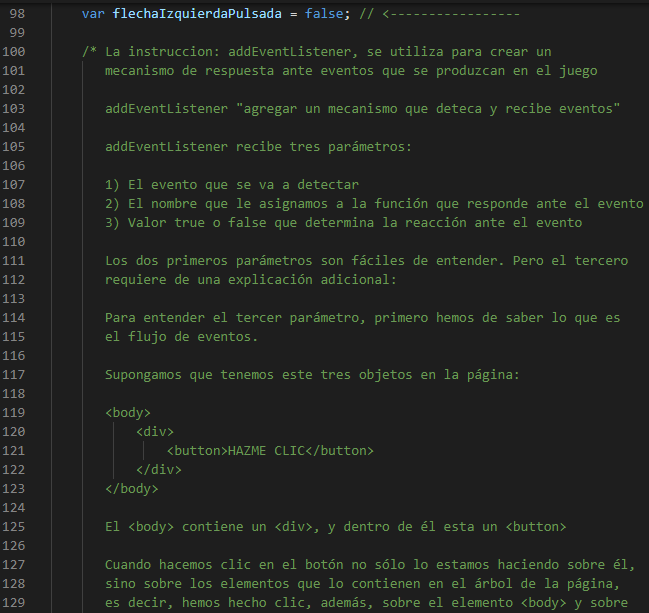
El paso número 3 consiste en crear la paleta en la cual la bola rebotará, la cual estará situada en la parte inferior del juego y será controlada por ambas flechas del teclado.

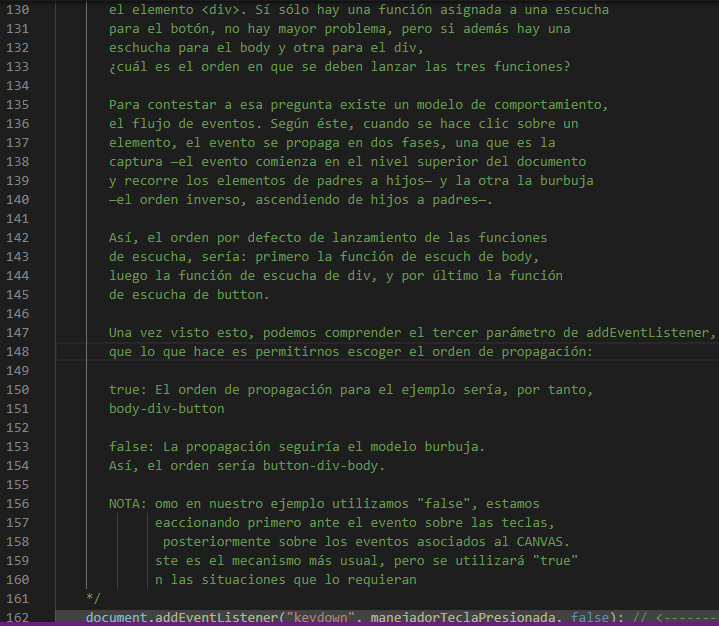
Para empezar con el tercer paso primero se deben crear dos variables a las cuales se les asignara el movimiento de la paleta con las flechas del teclado. Estas variables llevaran el siguiente nombre flechaDerechaPulsada y flechaIzquierdaPulsada. Luego de esto se crea una función la cual maneja el movimiento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

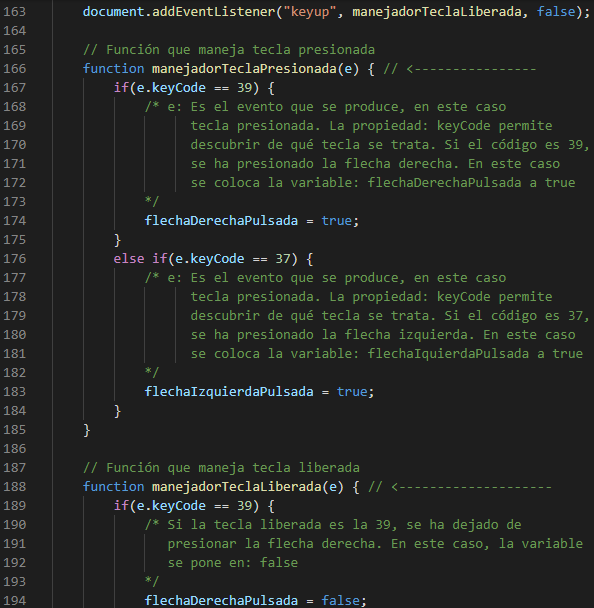


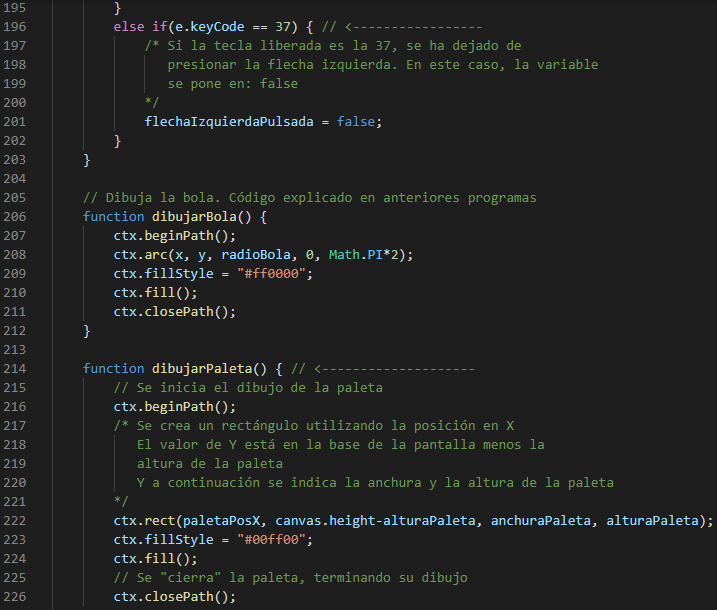


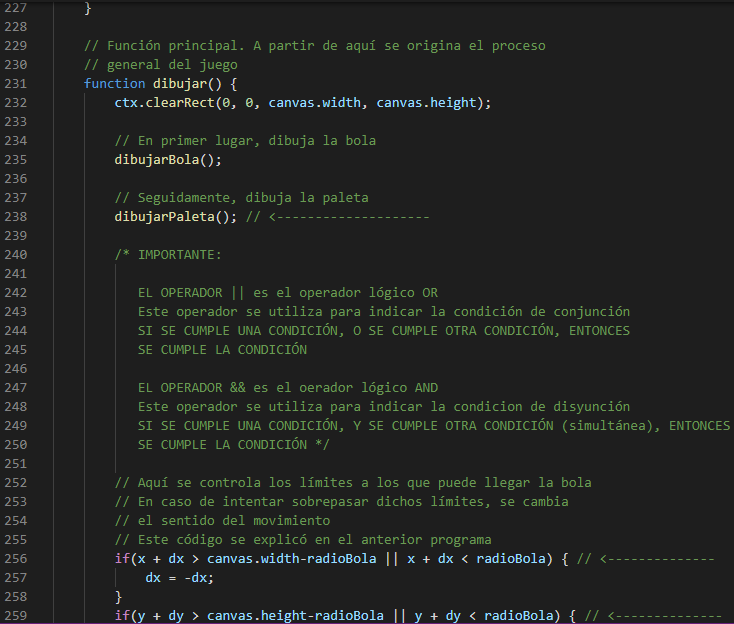


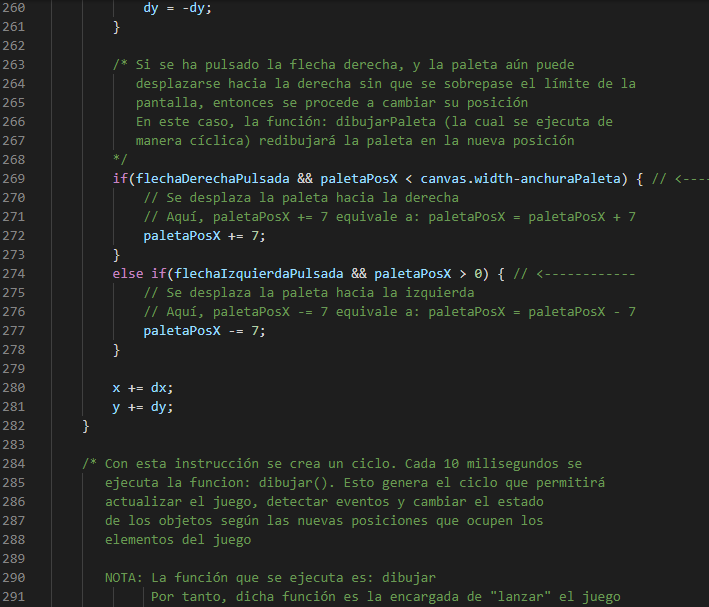


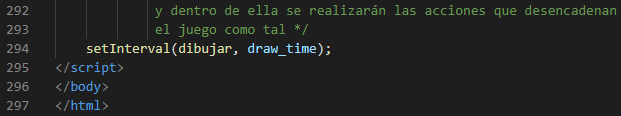




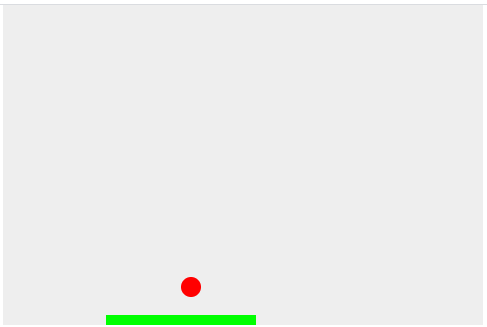








Al ejecutar este código se obtiene la siguiente interfaz visual:



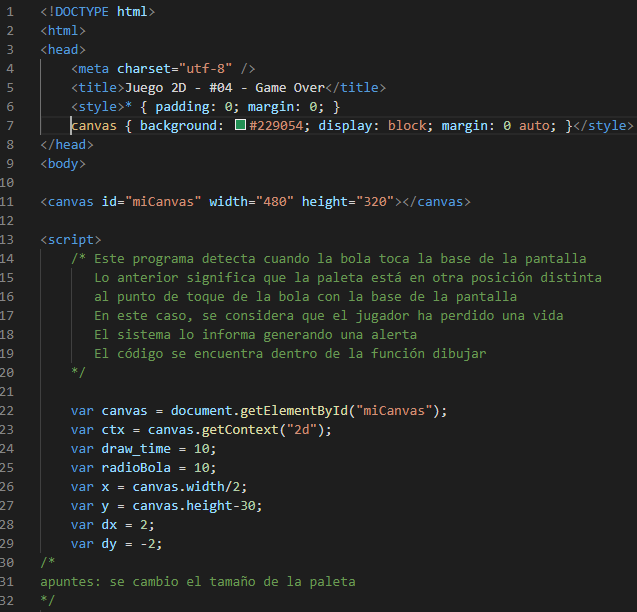
En la figura 4 podemos observar la bola y la paleta en la parte inferior del juego creadas anteriormente en la parte número 3 del código.

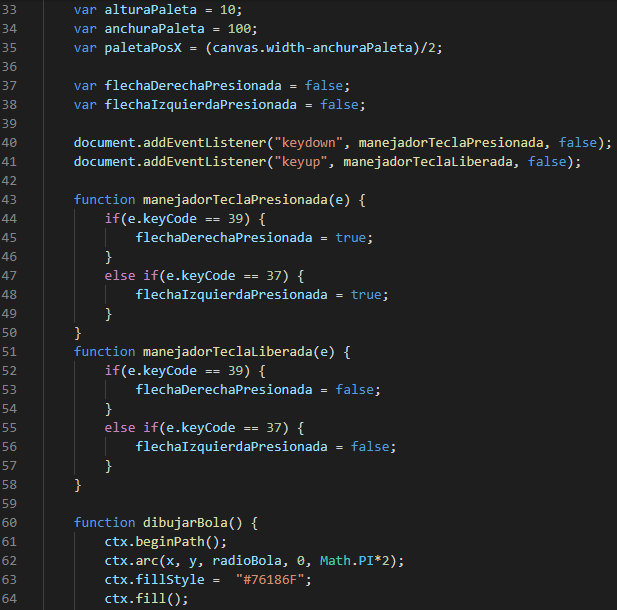
En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

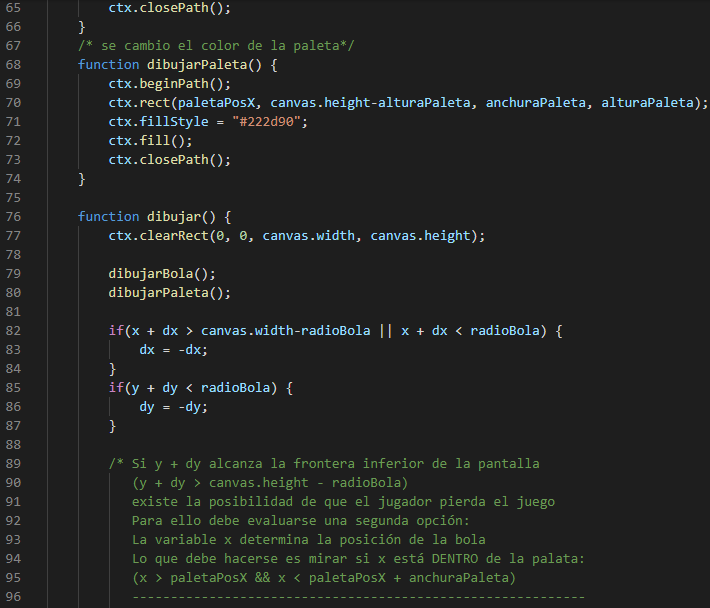
# FASE 4: Fin del juego

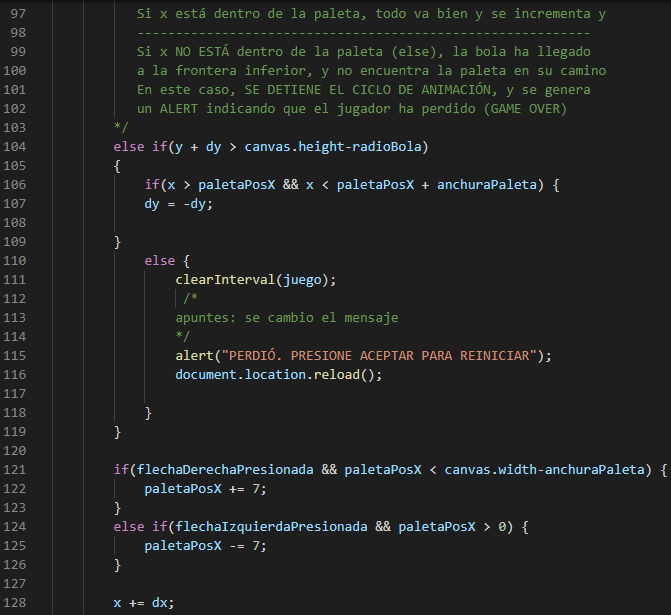
En esta parte del programa programaremos que se pueda detectar cuando la bola toca la base de la pantalla, en una coordenada diferente a la de donde se encuentra la paleta, lo que hará que el juego se pierda.

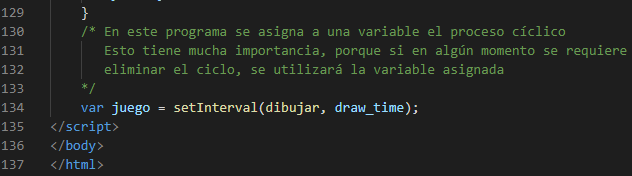
Para este caso analizaremos un código en la función dibujar, el codigo seria: (y + dy > canvas.height - radioBola) el cual se utilizaría para cuando la bola toque la parte inferior del juego lo cual haría que el juego se pierda. Pero para estar seguros de que el juego se ha perdido analizaremos el siguiente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



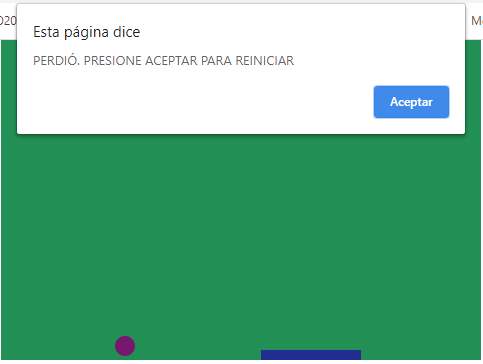








Al ejecutar este código se obtiene la siguiente interfaz visual:



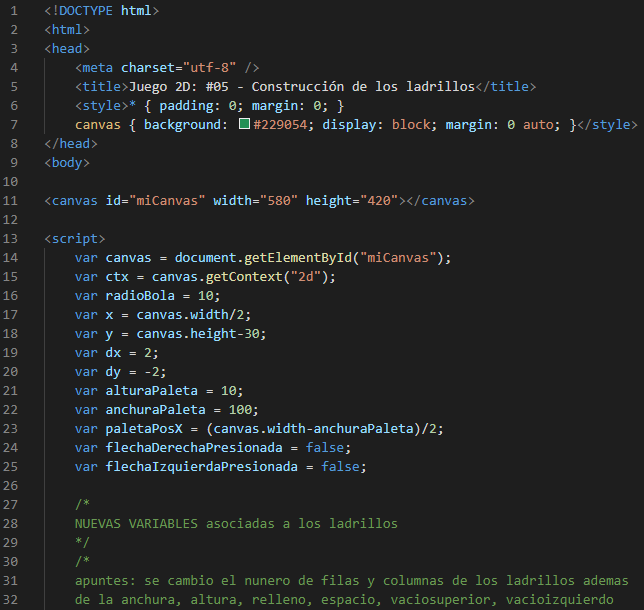
En la figura 5 podemos observar como la bola al tocar la parte inferior del juego y al estar en una coordenada diferente a la paleta aparece perdió. Presione aceptar para reiniciar que significa que el juego se ha perdido y se ha acabado

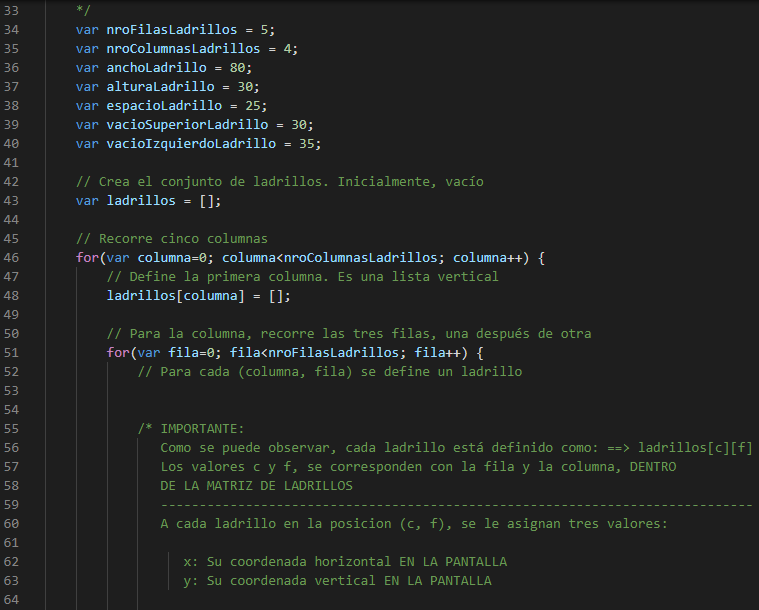
En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

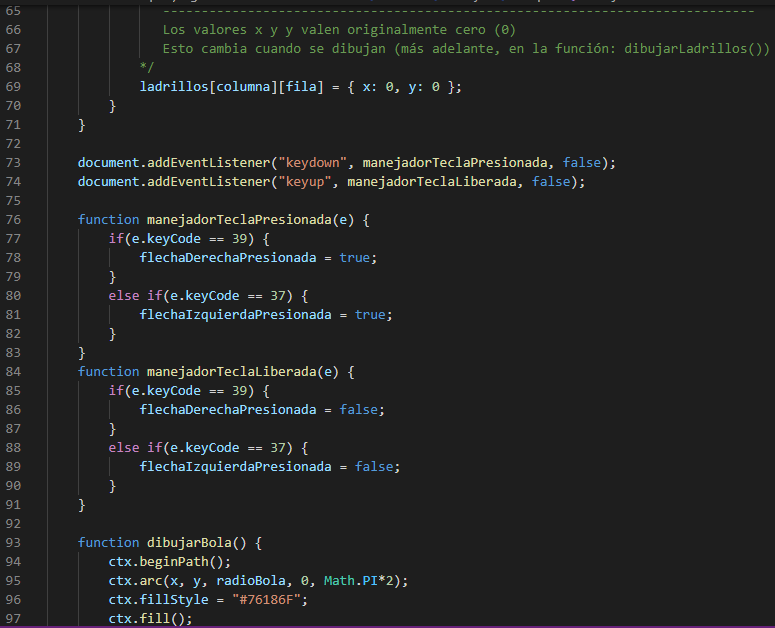
# FASE 5: Muro de ladrillos

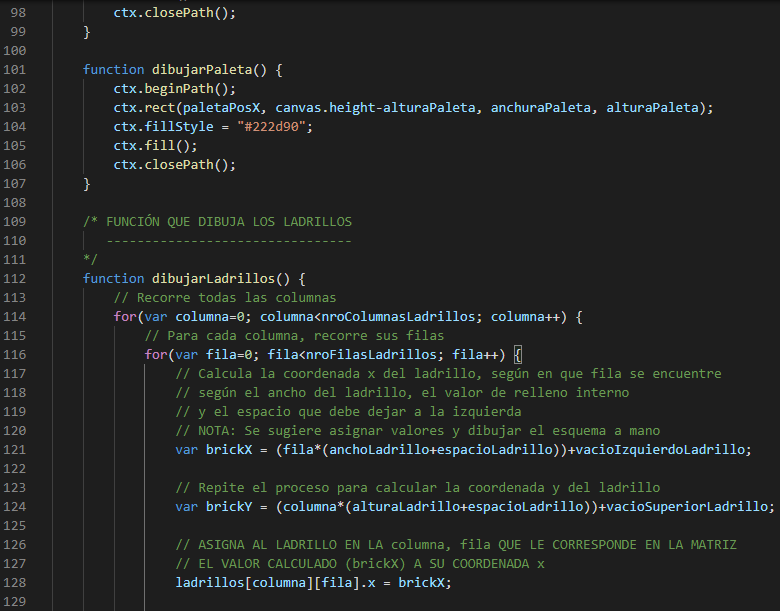
En esta parte del juego crearemos unas variables las cuales crearan un muro de ladrillos dentro del juego en los cuales rebotara la bola.

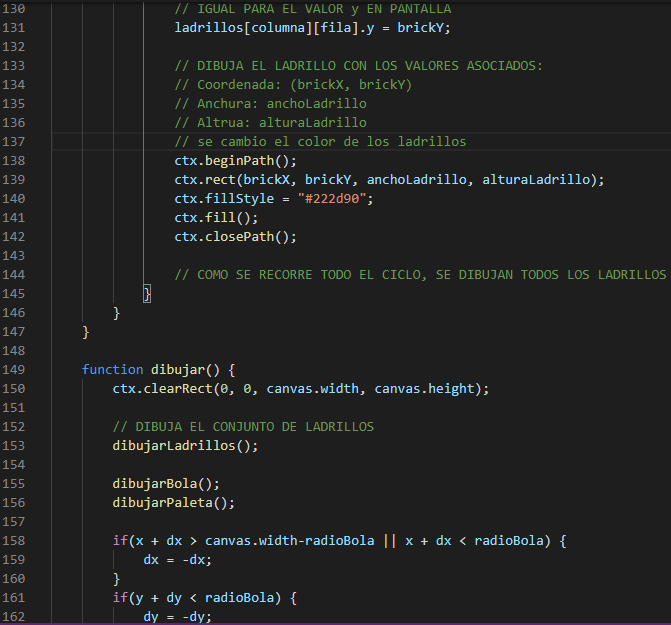
Analizaremos la sgte funcion: function dibujarLadrillos() , esta función se apoya de varias variables para la creación del muro de los ladrillos la cual la hace analizando la columna y la fila en la que quedara asignado cada ladrillo.

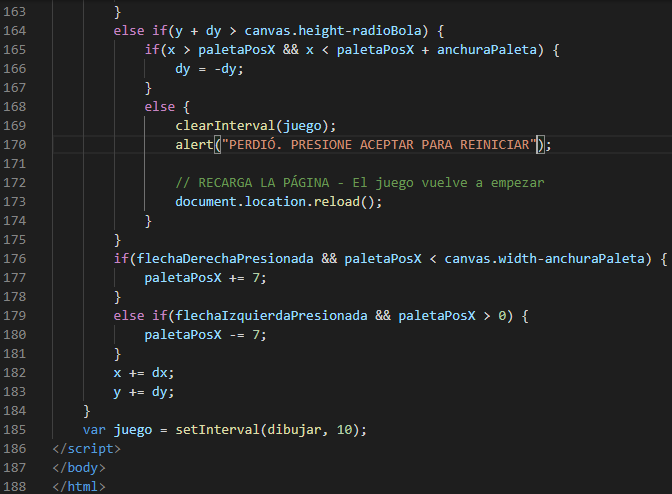




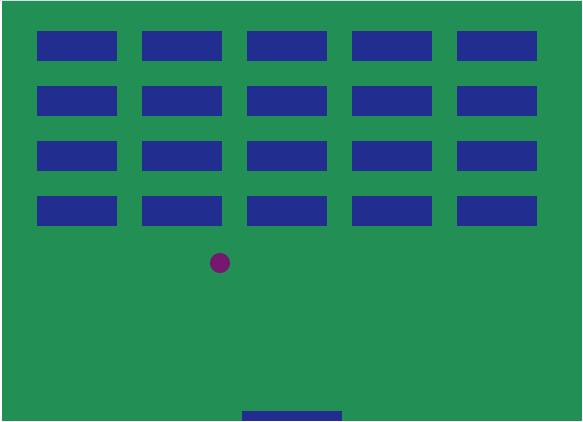








Al ejecutar este código se obtiene la siguiente interfaz visual:



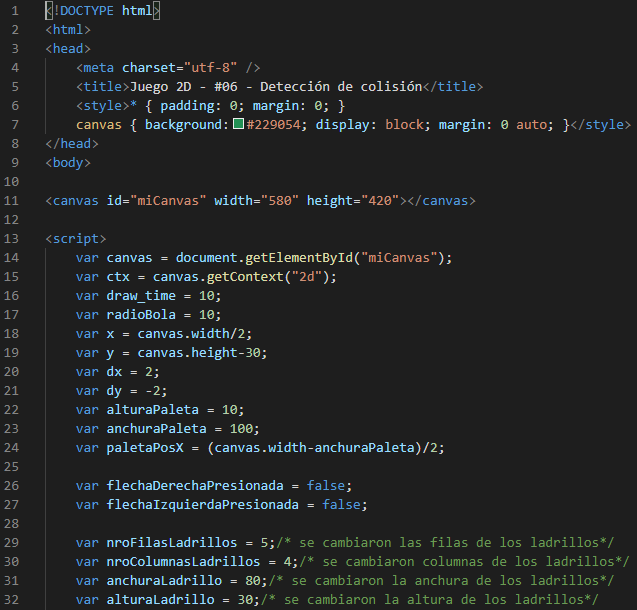
En la figura 6 podemos observar la creación de la pared de ladrillos dentro del campo del juego. Cabe recalcar que los ladrillos aún son traspasables para la bola

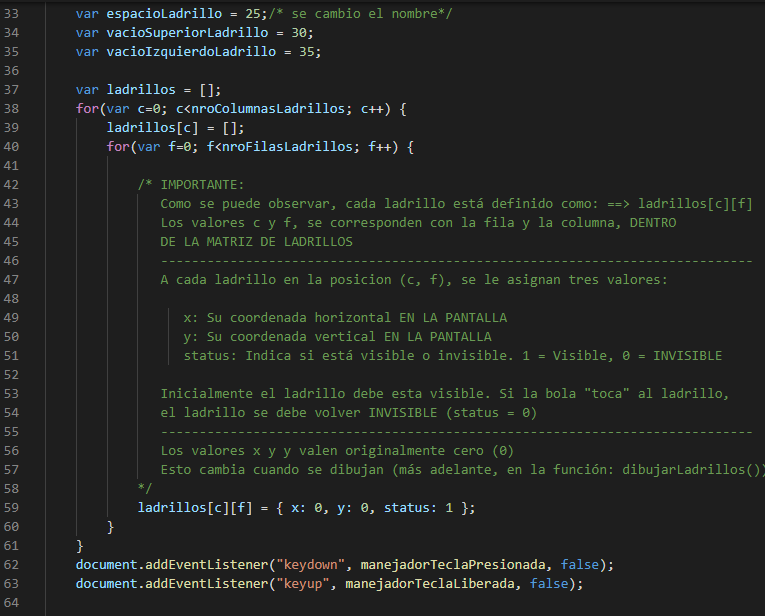
En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

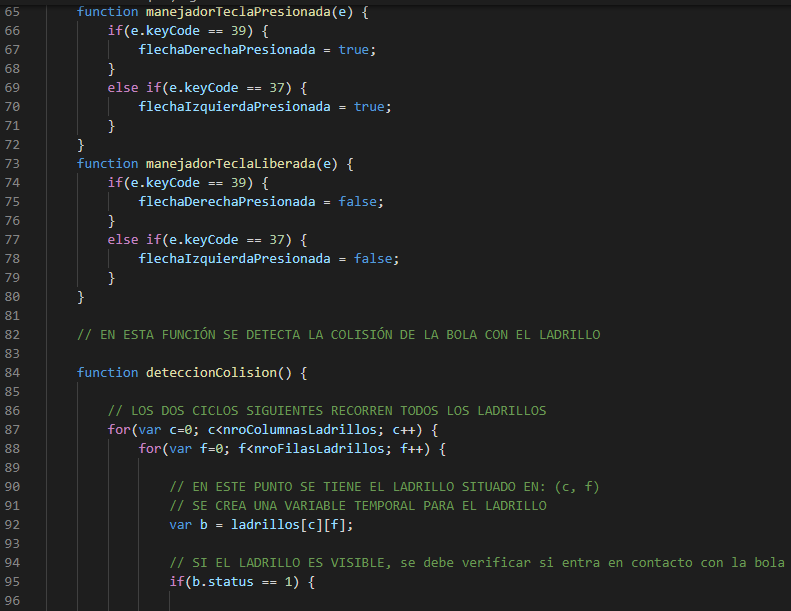
# FASE 6: Detección de colisiones

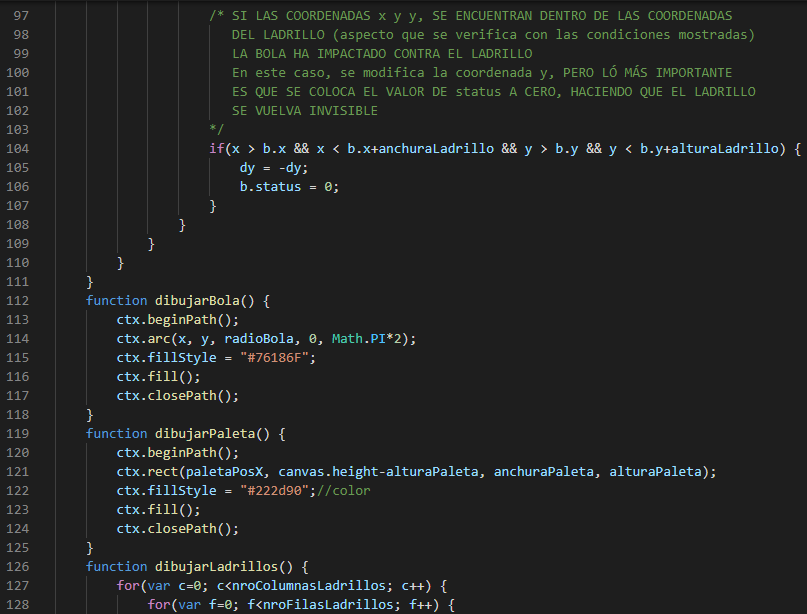
En esta parte del programa realizaremos la función que hará que se detecte la colisión de la bola con alguno de los ladrillos y al ocurrir esto hara que el ladrillo con el que colisiono desaparezca.

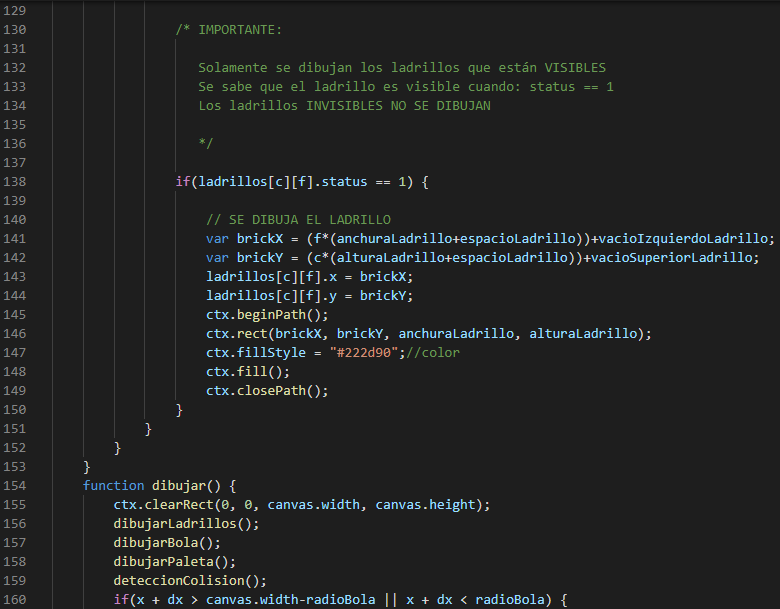
Procederemos a analizar la función que hace esto posible, la función será la siguiente: function deteccionColision(), esta es la función que permite que cuando la bola colisione con alguno de los ladrillos desaparezca, esto se realiza creando una variable temporal en la cual se asigna el ladrillo y analizando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

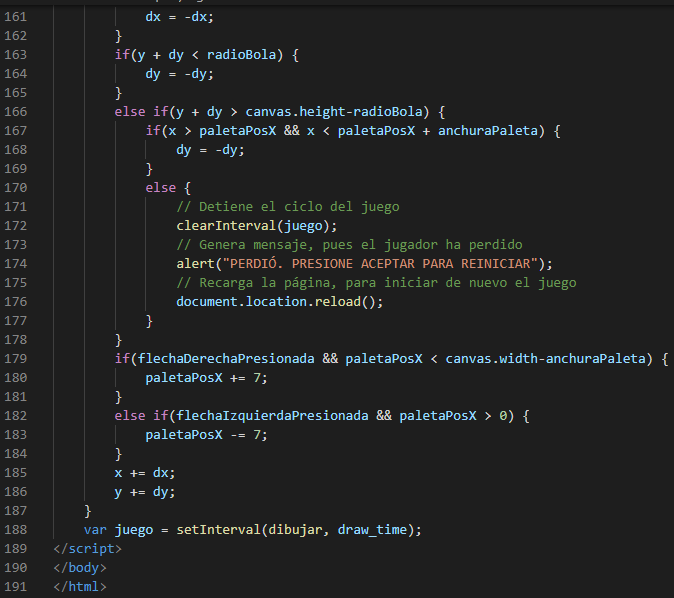




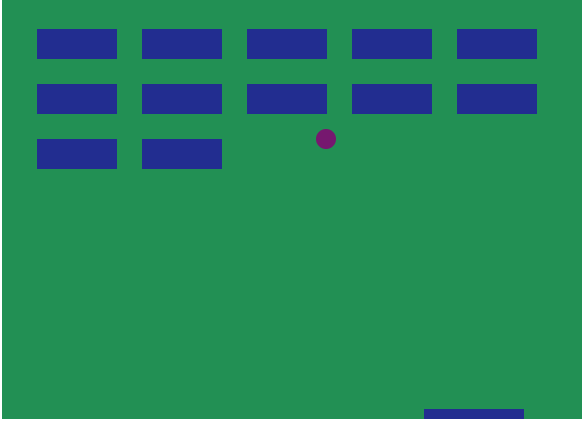








Al ejecutar este código se obtiene la siguiente interfaz visual:



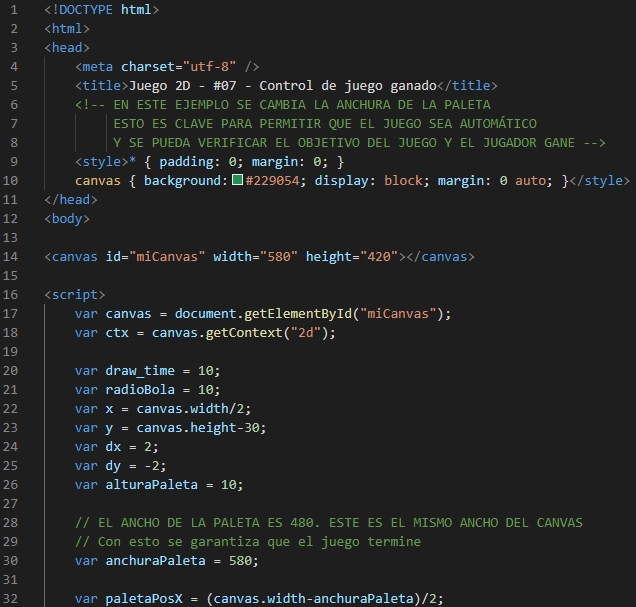
En la figura 7 podemos observar cómo algunos ladrillos se desaparecieron luego de ser golpeados por la bola.

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

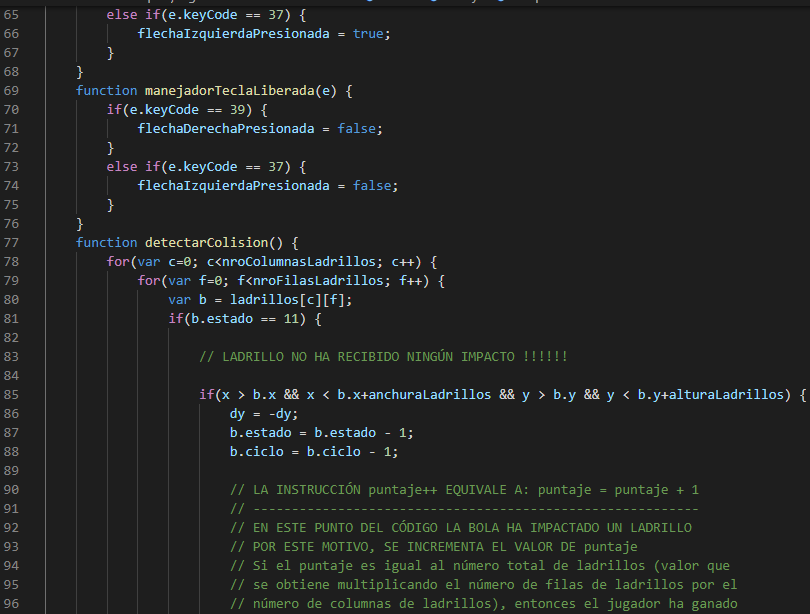
# FASE 7: Contar puntos y ganar

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

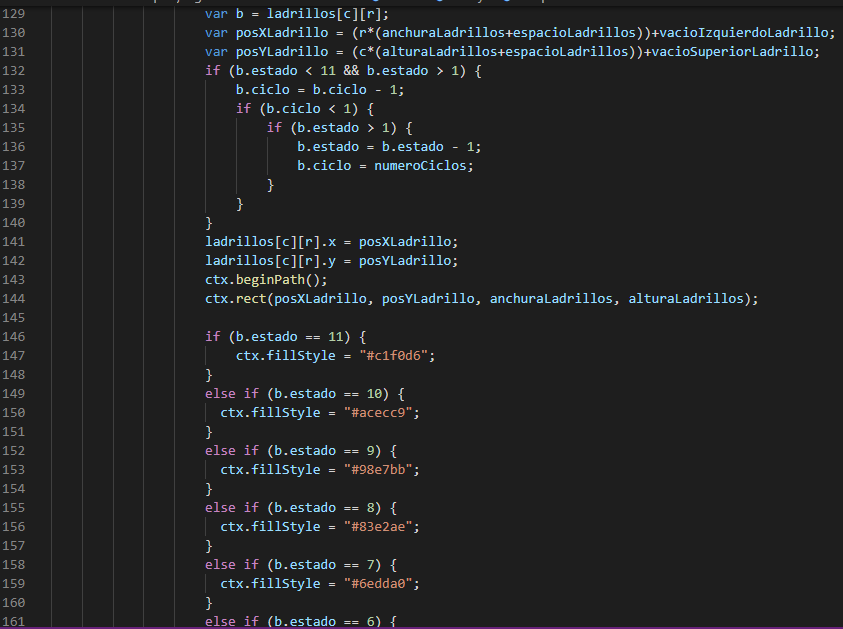
Se crea una variable llamada puntaje la cual controla la cantidad de ladrillos que han sido golpeados por la bola, cada que la bola impacta un ladrillo se le agrega un valor a esta variable hasta que el puntaje es igual al número de ladrillos haciendo que el juego se gane. Además, se le agrega un efecto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

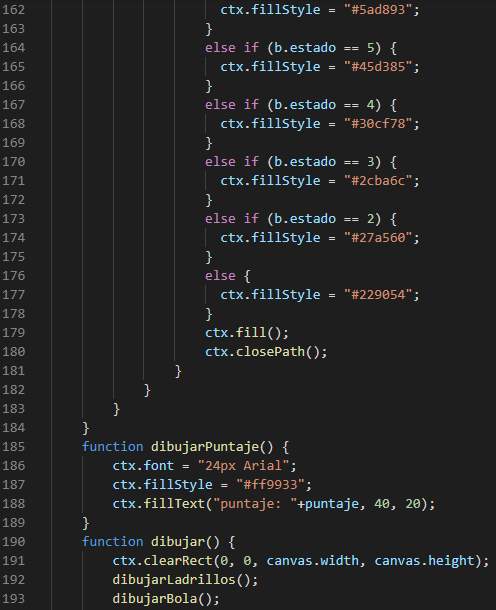


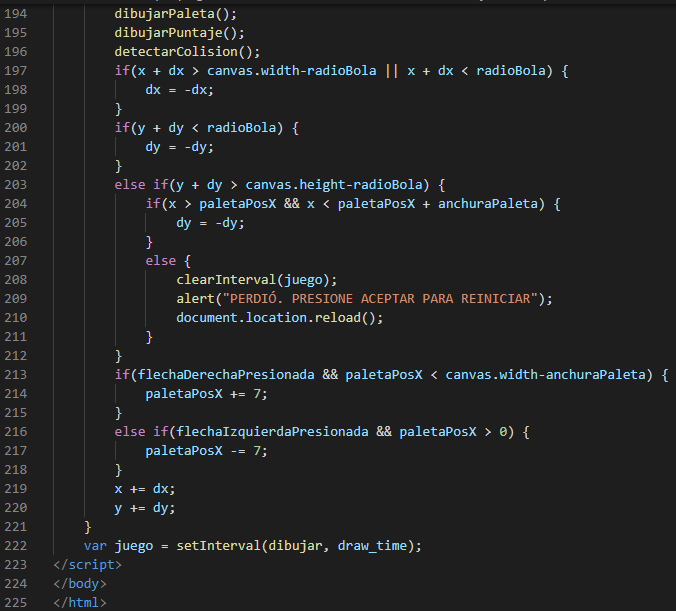




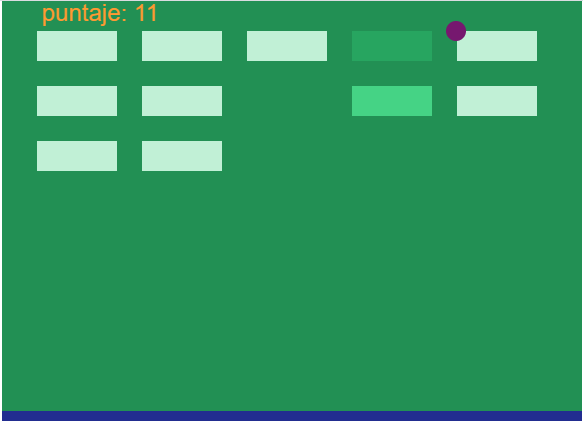




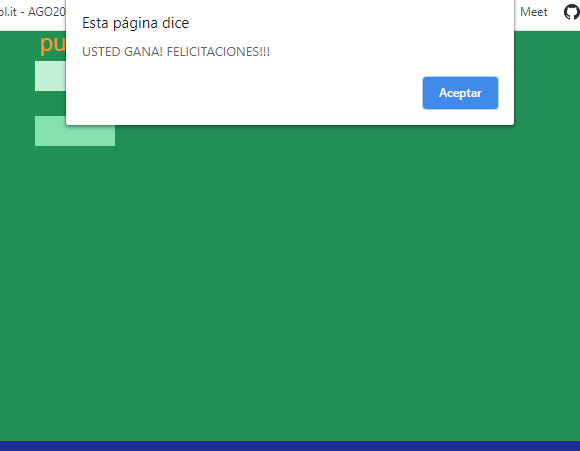




Al ejecutar este código se obtiene la siguiente interfaz visual:



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.

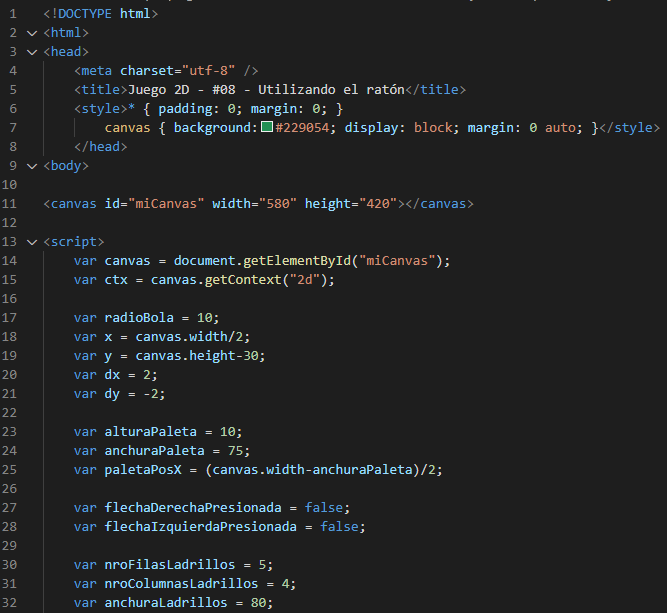


En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

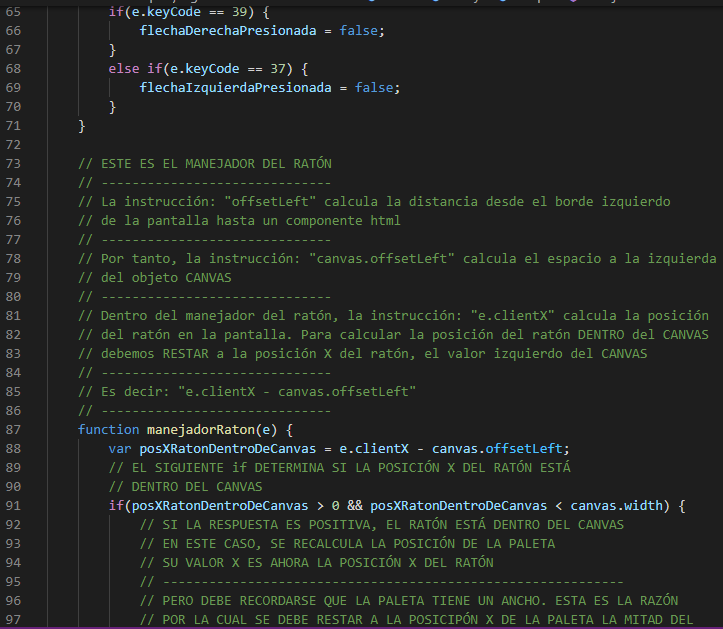
# FASE 8: Controlando el ratón

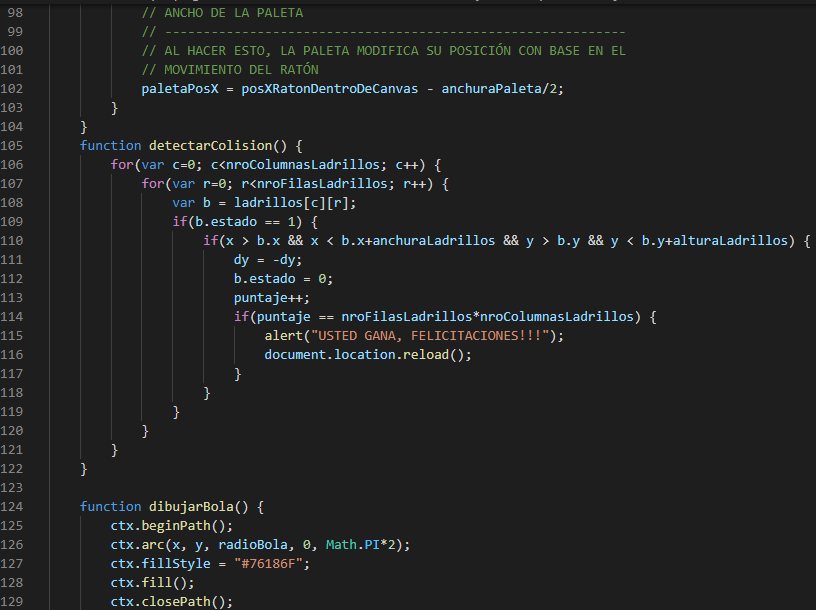
En esta parte del programa haremos que la paleta en lugar de ser movida por las flechas sea movida por el mouse.

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.

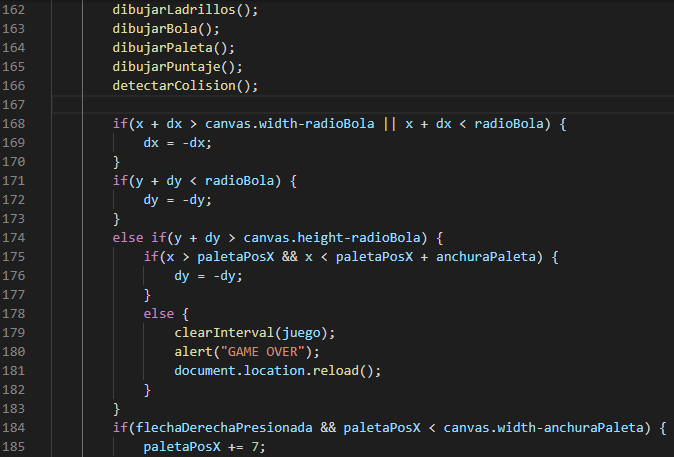


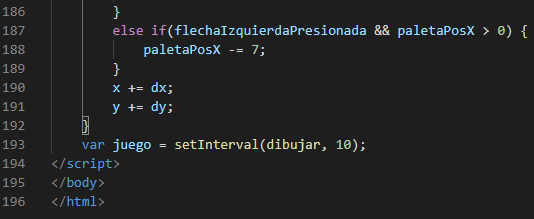












Al ejecutar este código se obtiene la siguiente interfaz visual:



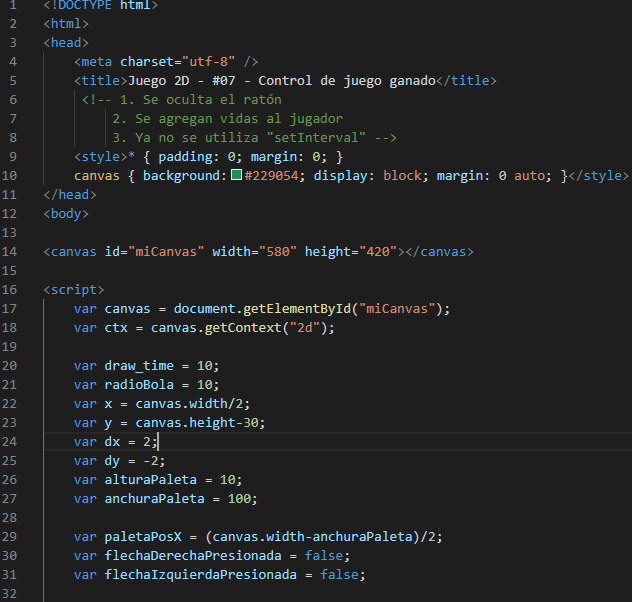
En la figura 9 se puede observar como la paleta es controlada de derecha a izquierda mediante el uso del mouse

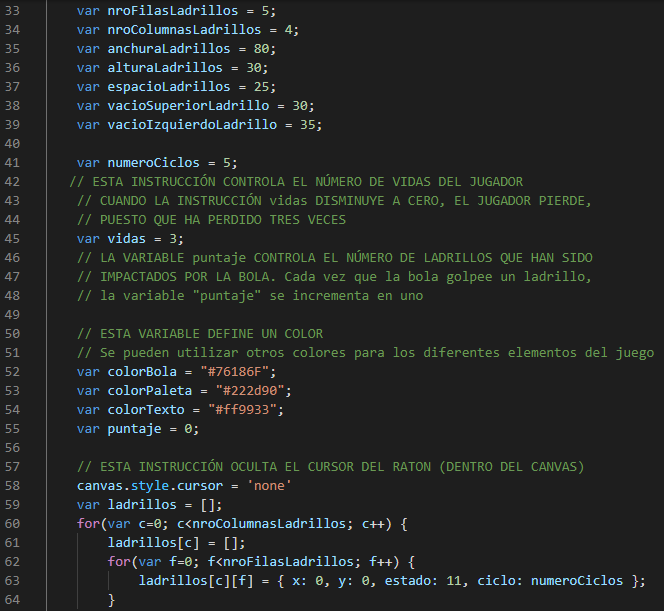
En el siguiente apartado se explicará la siguiente fase del juego. En caso de ser necesario, se agregarán todas las explicaciones que sean necesarias para que el juego quede debidamente explicado.

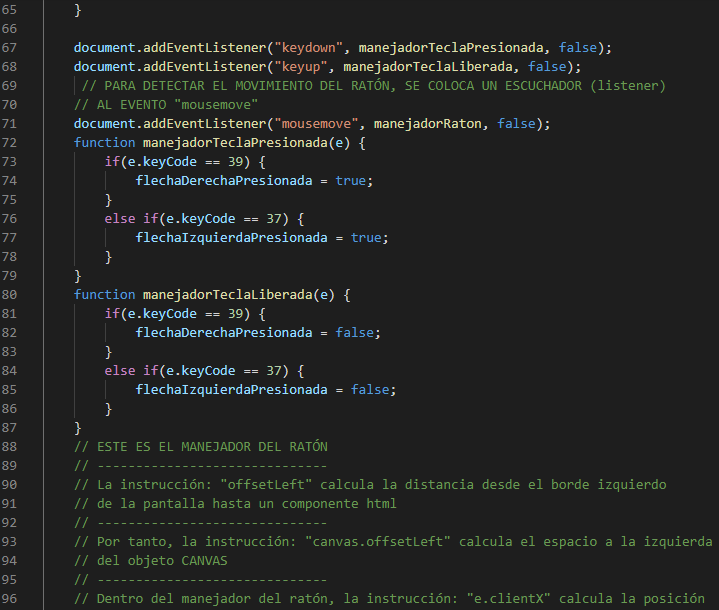
# FASE 9: Finalizando el juego

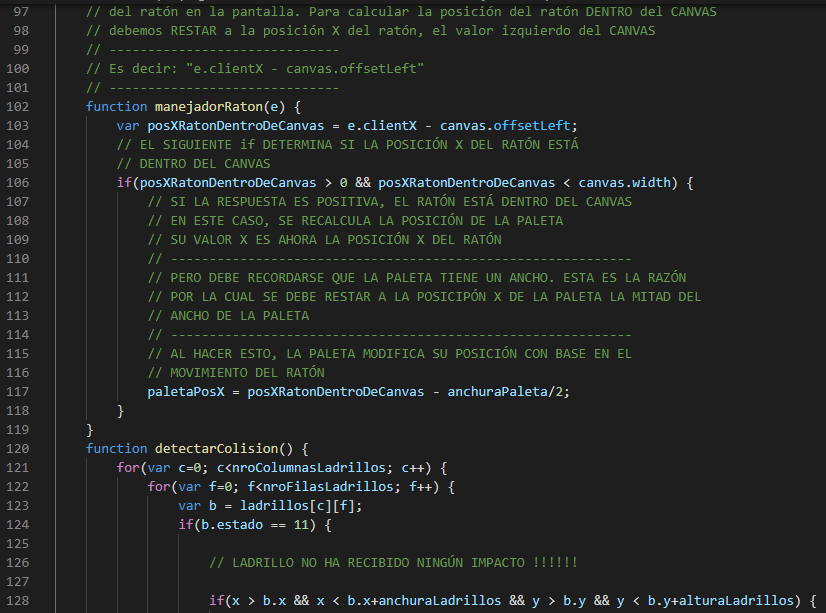
En esta parte del programa ya se agregan los toques finales tales como vidas para el jugador, ocultar el mouse entre otras.

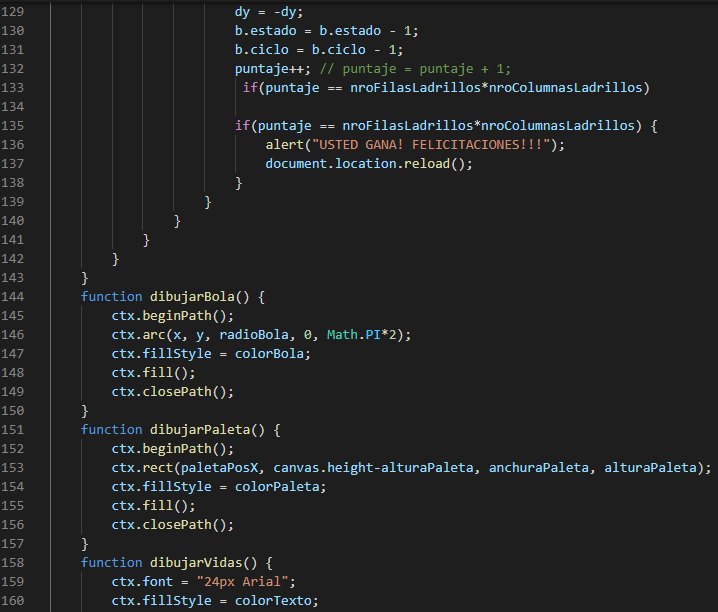
Se crea una variable var vidas = 3 con la instrucción de controlar las vidas que tiene dentro del juego cada participante y se crea otra variable canvas.style.cursor = 'none' para ocultar el mouse dentro del campo del juego, también se crea la instrucción vidas--; la cual lleva la cuenta de las vidas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



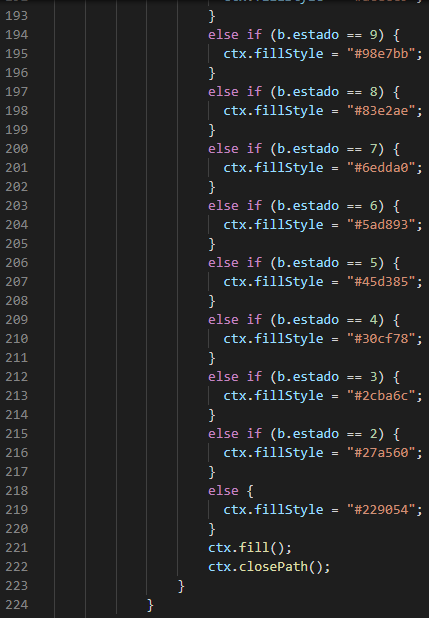


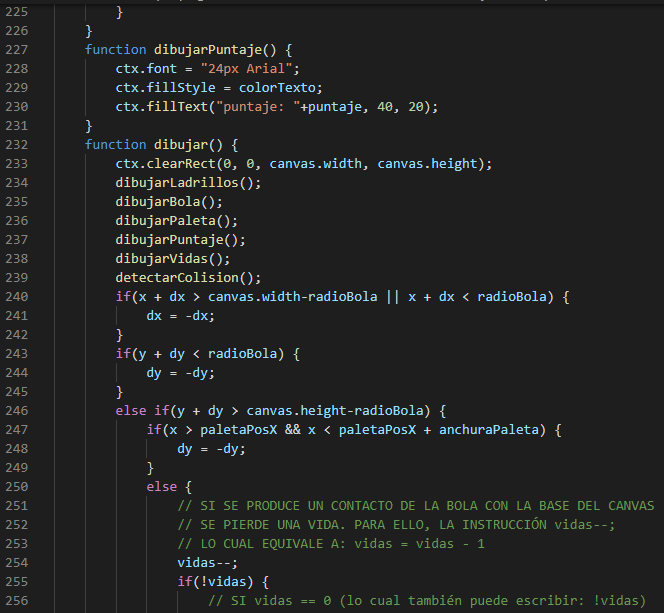


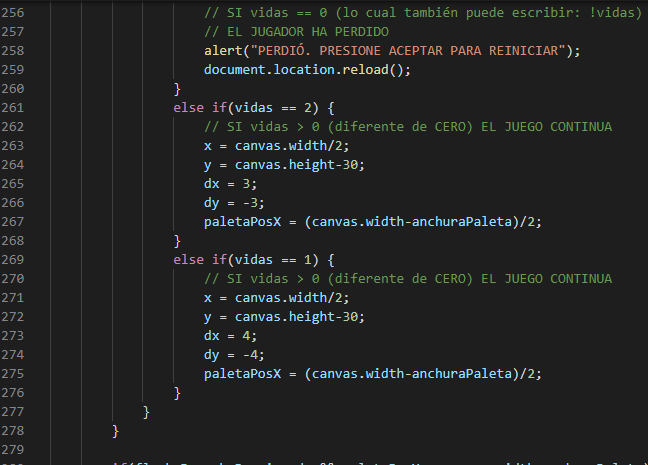


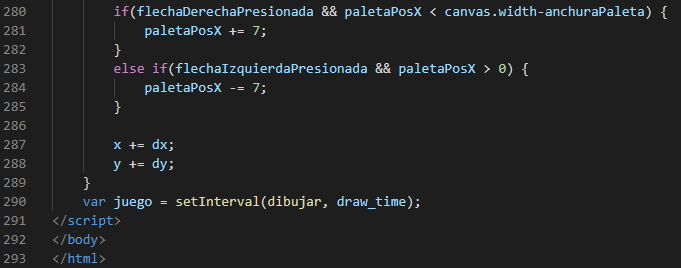




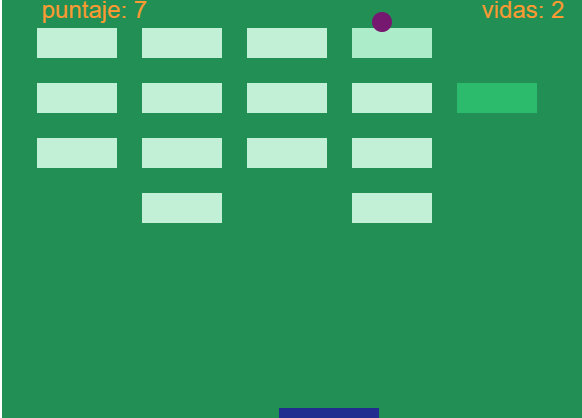








Al ejecutar este código se obtiene la siguiente interfaz visual



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.

# CONCLUSIONES

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

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

# BIBLIOGRAFÍA

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