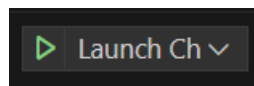


Capturas de pantalla explicando el proceso de Debug

Para empezar, al comenzar a debugear se nos generará un archivo json al que tendremos que cambiar la url que viene por defecto por la de nuestro navegador en cuestión.

```
"type": "chrome",  
"request": "launch",  
"name": "Launch Chrome against localhost",  
"url": "http://127.0.0.1:5500/",  
"webRoot": "${workspaceFolder}"
```

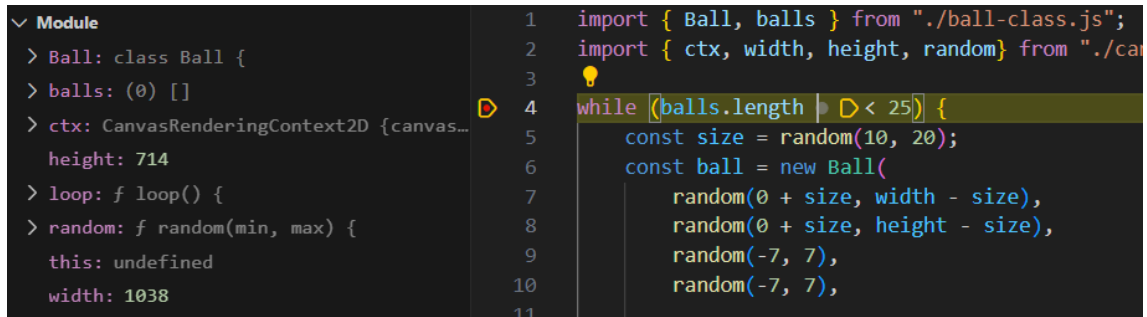
Tras esto, comenzaremos a debugear pulsando en la flechita verde que nos aparece en la esquina superior izquierda de la pantalla.



Una vez hecho esto, podemos comenzar a poner puntos de ruptura. En este caso, debemos hacerlo en el bucle iterativo que genera las bolas de este ejercicio e ir viendo cómo se generan, además de su color aleatorio y sus diferentes atributos.

```
3  
4  while (balls.length < 25) {  
5      const size = random(10, 20);  
6      const ball = new Ball(  
7          random(0 + size, width - size),  
8          random(0 + size, height - size),  
9          random(-7, 7),  
10         random(-7, 7),  
11         size  
12     );  
13     balls.push(ball);  
14  
15  
16 }  
17
```

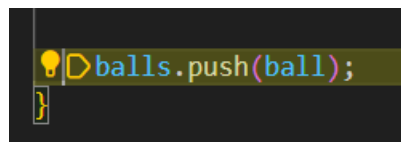
Ahora pasaremos a comprobar como se van creando las bolas, pudiendo ver ahora que tenemos 0 creadas, como nos indica el array de balls: (0) [] que se puede apreciar en la siguiente imagen.



```
Module
> Ball: class Ball {
> balls: (0) []
> ctx: CanvasRenderingContext2D {canvas: ..., height: 714}
> loop: f loop() {
> random: f random(min, max) {
  this: undefined
  width: 1038
}

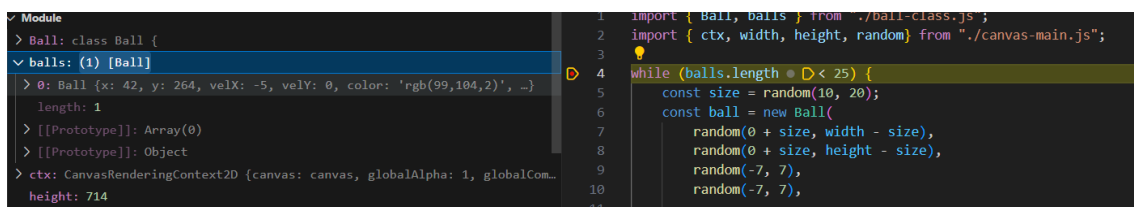
1 import { Ball, balls } from './ball-class.js';
2 import { ctx, width, height, random } from './canvas-main.js';
3
4 while (balls.length < 25) {
5   const size = random(10, 20);
6   const ball = new Ball(
7     random(0 + size, width - size),
8     random(0 + size, height - size),
9     random(-7, 7),
10    random(-7, 7),
11  )
}
```

Podemos ver como dejamos el punto de ruptura en la función que genera las bolas y el valor de estas sigue todavía a 0.



```
balls.push(ball);
}
```

Una vez que pasemos por aquí, se nos devolverá al comienzo del bucle, pudiendo ver que ya se ha creado una bola como nos indica el balls: (1) []. Además de esto, podemos ver los diferentes atributos de esta al desplegar la flechita que nos aparece a su izquierda.



```
Module
> Ball: class Ball {
> balls: (1) [Ball]
  length: 1
  0: Ball {x: 42, y: 264, velX: -5, velY: 0, color: 'rgb(99,104,2)', ...}
  [[Prototype]]: Array(0)
  [[Prototype]]: Object
> ctx: CanvasRenderingContext2D {canvas: canvas, globalAlpha: 1, globalCom...}
height: 714

1 import { Ball, balls } from './ball-class.js';
2 import { ctx, width, height, random } from './canvas-main.js';
3
4 while (balls.length < 25) {
5   const size = random(10, 20);
6   const ball = new Ball(
7     random(0 + size, width - size),
8     random(0 + size, height - size),
9     random(-7, 7),
10    random(-7, 7),
11  )
}
```

Antes de volver a ver como se genera otra bola, podemos ver que cuando nos situamos en la función que las crea nos da una “preview” de la bola que se va a crear, y tras volver al comienzo del bucle vemos como esta misma bola se ha creado y añadido.

This screenshot shows a code editor with a JavaScript file. On the left, a 'Block' panel displays the details of a selected object: a Ball with x: 325, y: 422, velX: -5, velY: 4, and color: 'rgb(63,106,78)'. Below this, the 'Module' panel shows the Ball class and the 'balls' array, which currently contains one Ball object. The 'WATCH' panel is empty. On the right, the code shows a while loop that creates a new Ball object and pushes it to the 'balls' array. The line 'balls.push(ball);' is highlighted.

This screenshot shows the same code editor after the second ball has been added. The 'Block' panel now shows the second Ball object with x: 42, y: 264, velX: -5, velY: 0, and color: 'rgb(99,104,2)'. The 'Module' panel shows the 'balls' array with two elements. The 'WATCH' panel still shows the canvas rendering context. The code on the right is the same, but the cursor is now at the start of the while loop.

Por último, realizamos la misma operación que con la bola anterior, siguiendo el mismo proceso y viendo como se crea con sus diferentes atributos y color aleatorios.

This screenshot shows the code editor after the third ball has been added. The 'Block' panel shows the third Ball object with x: 238, y: 396, velX: -7, velY: 6, and color: 'rgb(114,21,1)'. The 'Module' panel shows the 'balls' array with three elements. The 'WATCH' panel still shows the canvas rendering context. The code on the right is the same, but the cursor is now at the start of the while loop.

This screenshot shows the code editor after the fourth ball has been added. The 'Block' panel shows the fourth Ball object with x: 42, y: 264, velX: -5, velY: 0, and color: 'rgb(99,104,2)'. The 'Module' panel shows the 'balls' array with four elements. The 'WATCH' panel still shows the canvas rendering context. The code on the right is the same, but the cursor is now at the start of the while loop.