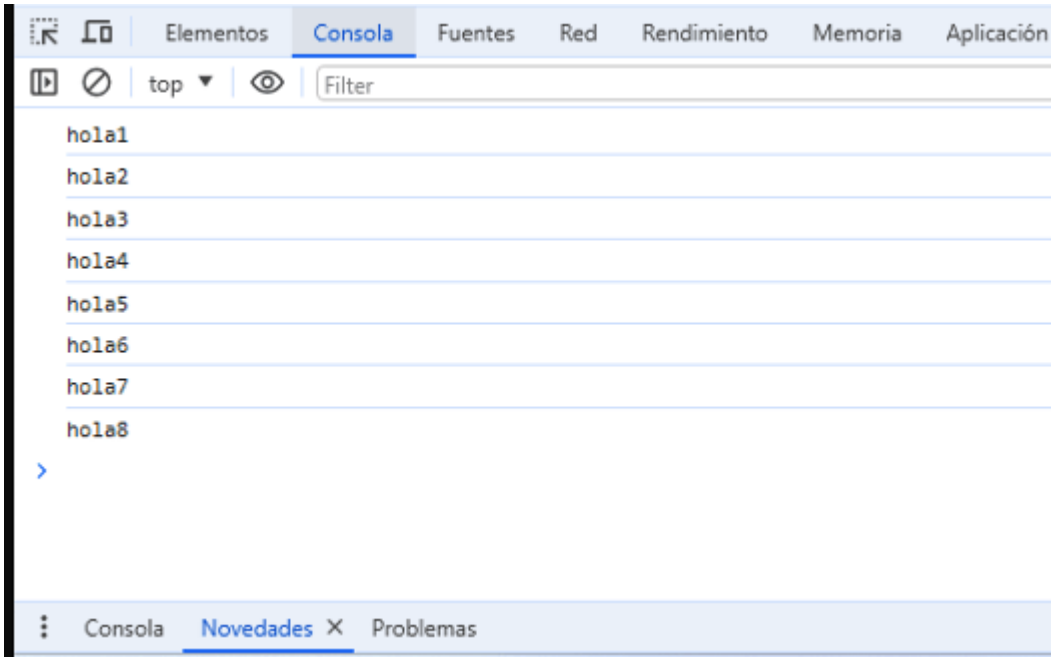
Un Web Worker es una característica de JavaScript que permite ejecutar scripts en segundo plano, es decir, en paralelo al hilo principal del navegador. Esto permite que las operaciones complejas o que consumen mucho tiempo se realicen sin bloquear la interfaz de usuario, lo que resulta en una experiencia más fluida y receptiva para el usuario. A muchas personas no les parece crítica esta funcionalidad pero lo es . Vamos a ver un ejemplo sencillo que explique la necesidad de un WebWorker con el siguiente código :

```
window.onload = function () {

        let numero = 0;
        setInterval(function () {
            numero++;
            console.log("hola" + numero);
        }, 2000)


    }
```

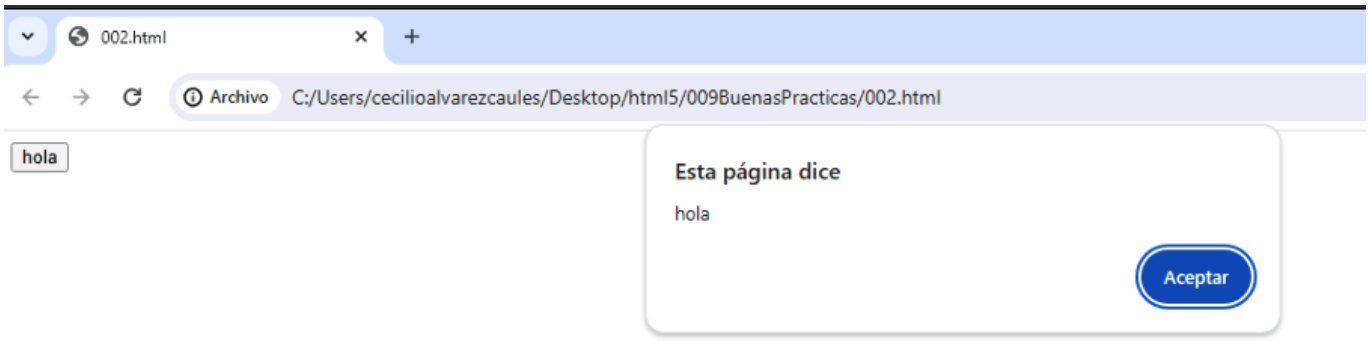
Este código cuando se ejecuta realiza una tarea cada 2 segundos (2000ms) que imprime un mensaje de hola por la consola . A muchas personas les parece que este es un código asíncrono.



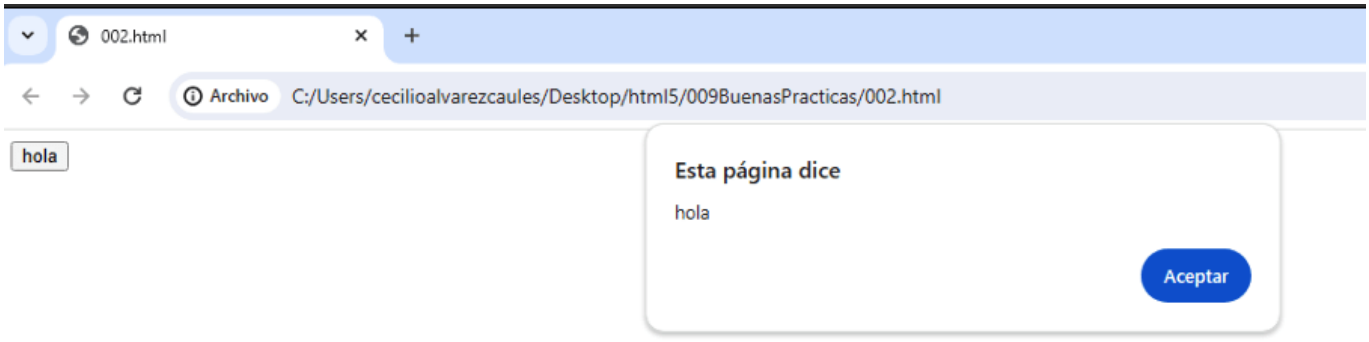
Realmente no lo es ya que JavaScript es mono hilo y lo que hace el hilo de ejecución principal es esperar 2 segundos hasta emitir el siguiente mensaje de la cola. ¿Que pasaría si cambiáramos el código por lo siguiente?.

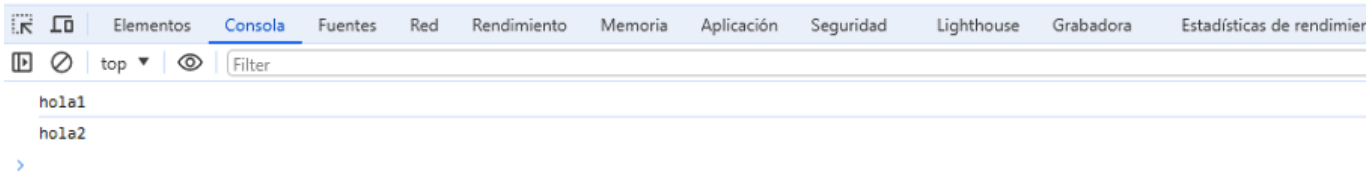
```
  window.onload = function () {

            let numero = 0;
            setInterval(function () {
                numero++;
                console.log("hola" + numero);
            }, 2000)

document.getElementById("hola").addEventListener("click",function() {

                alert("hola")
            })

        }
```

Se trata de añadir un evento de click a un botón de hola que tengamos en pantalla de tal forma que cuando pulsemos nos lance una ventana de alert.



Esto en principio no genera ningún problema ya que aparece el popup del alert de hola y poco más ahora bien . Si revisamos la consola nos daremos cuenta de que los mensajes dejan de avanzar y se congelan.





¿A que se debe esto? . Pues muy sencillo al ser JavaScript mono hilo nos encontraremos que hemos bloqueado el hilo con un pop up emergente. Ya no se puede ejecutar nada más . Si queremos que los mensajes de la consola avancen necesitamos pulsar en esta caja. Algo que no tiene mucho sentido

WebWorkers al rescate

¿Como podemos solventar esto? . Para solventarlo debemos construirnos un WebWorker. Un WebWorker nos permite ejecutar tareas de javascript en paralelo. Para ello es tan sencillo como definir un fichero js que contenga el código que queremos ejecutar.

```
let numero = 0;
setInterval(function () {
    numero++;
    console.log("hola" + numero);
}, 2000)
```

Una vez hecho esto nos queda ver como queda nuestra nueva versión de código:

```
window.onload = function () {

            var worker= new Worker("miworker.js");
document.getElementById("hola").addEventListener("click",function() {

                alert("hola")
            })

        }
```

Ahora las dos cosas se ejecutan de forma paralela nosotros podremos quedarnos pendientes de pulsar al popup de alert . Pero el Webworker se encargará de la tarea de ejecutar las diferentes lineas de mensaje de la consola

Otros artículos relacionados

- Servlet 3.0 (II) Servlets Asincronos
- Introducción a JMeter y pruebas de carga
- JavaScript Console y otros métodos
- Java Runnable y Lambdas
- El concepto de Java infinite Stream