Depurar código desde el debugger del navegador

Ejemplo

Usaremos un ejemplo de código en el que el usuario introduce 2 números desde un formulario y imprimimos el resultado. El resultado de 2+3 debería de ser 5 pero nos muestra 23 debido a que no hemos convertido los valores del formulario a números (por defecto son String)

Como acceder al debugger

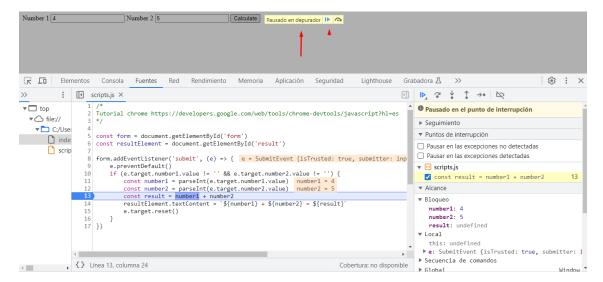
Para empezar a debuggear abrimos el navegador en este caso Chrome pero en Firefox es muy similar -> pulsamos en **f12** o **menú contextual inspeccionar**. Nos vamos a la pestaña de **fuentes** y buscamos nuestro **archivo js**.



Breackpoints

Los puntos de ruptura o breackpoints sirven para detener la ejecución del código en un punto que queramos para añadir uno pulsaremos en el número de la línea a la que queremos detener el código:

```
scripts.js ×
                                                                                          F
   Tutorial chrome https://developers.google.com/web/tools/chrome-devtools/javascript?hl=e
 3 */
 4
 5 const form = document.getElementById('form')
   const resultElement = document.getElementById('result')
 8 form.addEventListener('submit', (e) => {
9
       e.preventDefault()
10
       if (e.target.number1.value != '' && e.target.number2.value != '') {
           const number1 = parseInt(e.target.number1.value)
11
           const number2 = parseInt(e.target.number2.value)
           const result = number1 + number2
           resultElement.textContent = `${number1} + ${number2} = ${result}`
14
15
           e.target.reset()
16
17 })
```



Esta caja amarilla significa que el programa se ha parado aquí justo en el el breackpoint tenemos un botón para ejecutar la siguiente línea o para ejecutar el programa de forma normal.

Si ponemos el cursor encima de las variables/constantes podemos ver el contenido que tienen

Al pasar a la siguiente línea con el botón nos aparece el contenido de la variable

Seguimiento o expresiones vigiladas(Firefox)

En esta pestaña podemos acceder a cualquier valor de las variables declaradas o también podríamos usar funciones como typeoff en el momento en el que estamos en un breackpoint

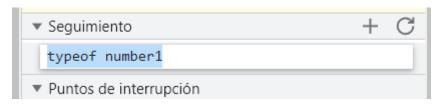
Ejecución: ponemos el nombre de una variable



Resultado:



Ponemos el nombre de una función de js separado nombre de variable

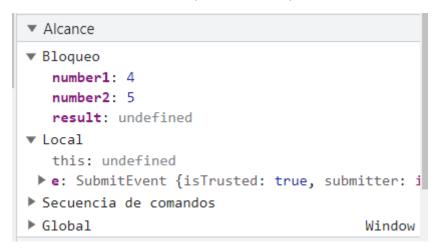


Resultado:



Alcance o ámbitos

Nos indica todas las variables que tenemos disponibles



Puntos de interrupción del procesador de eventos o puntos de ruptura de escucha de eventos (Firefox)

Podemos hacer puntos de ruptura en prácticamente en cualquier evento clicks, teclas, animaciones etc.

7	Puntos de interrupción del procesador de eventos
	→ ☐ Analizar
→	Animación
→	Arrastrar/soltar
⊩	Canvas
⊩	Cargar
⊩	► Control
⊩	Dispositivo
!	Geolocalización
-	▶ 🔲 Imagen en imagen
 	▶ ☐ Multimedia
•	Mutación DOM
 	▶ □ Notificación
•	Portapapeles
 	Puntero
٠,	Ratón
	☐ auxclick
	✓ click

Hay más cosas que aprender sobre el debugger pero con esto ya podemos empezar a trabajar, no importa en que navegador lo utilicemos.