

VALIDACION Y DEBUGGER DE CODIGO JAVASCRIPT





Fabian Florian

ING. SISTEMAS - MENTOR BOOTCAMP DWFS



JavaScript

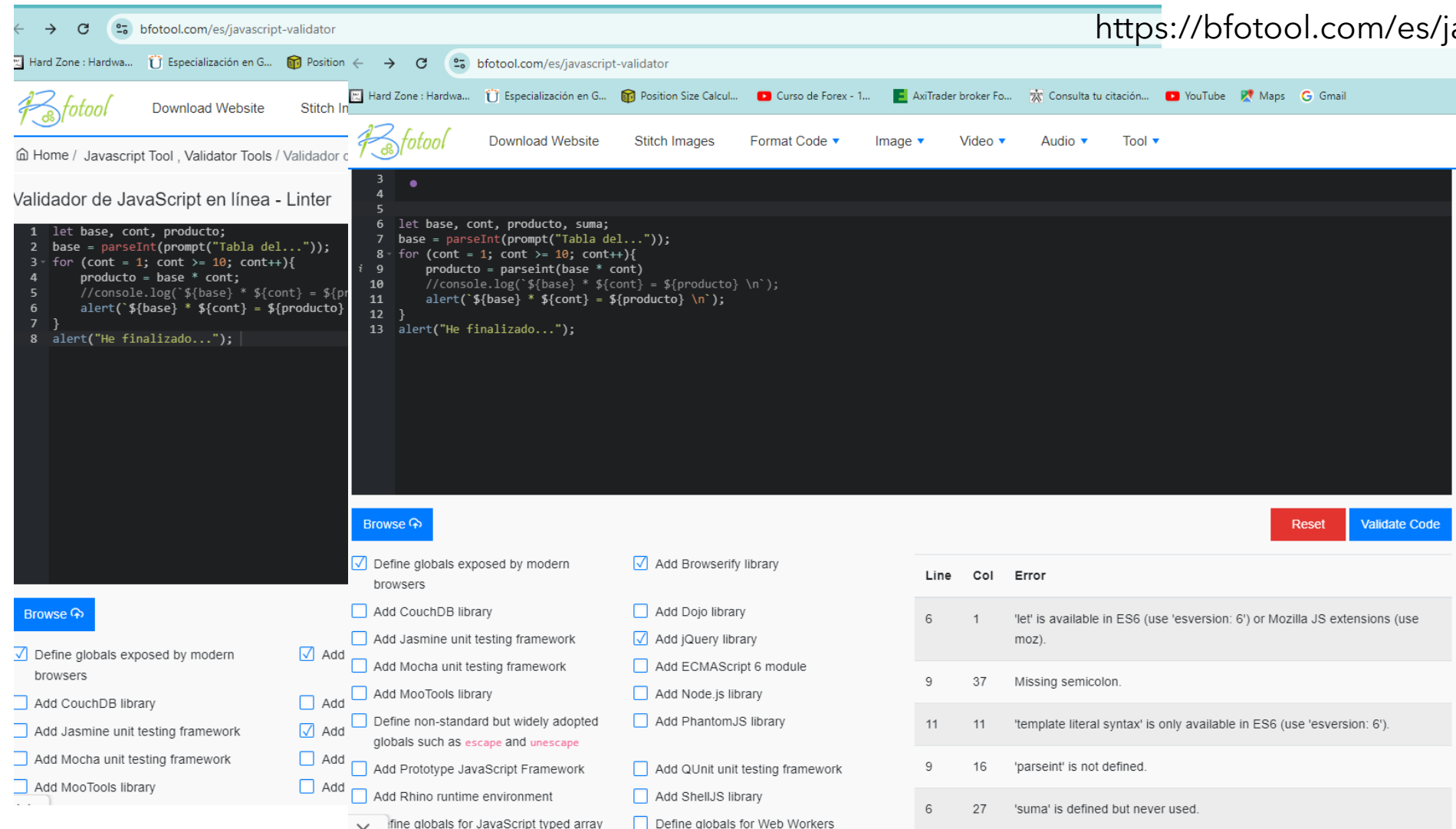


**VALIDANDO Y
DEPURANDO
CODIGO
JAVASCRIPT**



VALIDAR CODIGO JAVASCRIPT (1)

<https://bfotool.com/es/javascript-validator>



The screenshot shows the bfotool.com/es/javascript-validator website. The main area is a code editor with the following JavaScript code:

```
3
4
5
6 let base, cont, producto, suma;
7 base = parseInt(prompt("Tabla del..."));
8 for (cont = 1; cont >= 10; cont++){
9     producto = parseInt(base * cont)
10    //console.log(`${base} * ${cont} = ${producto} \n`);
11    alert(`${base} * ${cont} = ${producto} \n`);
12 }
13 alert("He finalizado...");
```

Below the code editor, there are several checkboxes for configuration:

- ☒ Define globals exposed by modern browsers
- ☐ Add CouchDB library
- ☐ Add Jasmine unit testing framework
- ☐ Add Mocha unit testing framework
- ☐ Add MooTools library
- ☐ Define non-standard but widely adopted globals such as `escape` and `unescape`
- ☐ Add Prototype JavaScript Framework
- ☐ Add Rhino runtime environment
- ☒ Add Browserify library
- ☐ Add Dojo library
- ☒ Add jQuery library
- ☐ Add ECMAScript 6 module
- ☐ Add Node.js library
- ☐ Add PhantomJS library
- ☐ Add QUnit unit testing framework
- ☐ Add ShellJS library
- ☐ Define globals for Web Workers

On the right side, there is a table of errors:

Line	Col	Error
6	1	'let' is available in ES6 (use 'esversion: 6') or Mozilla JS extensions (use moz).
9	37	Missing semicolon.
11	11	'template literal syntax' is only available in ES6 (use 'esversion: 6').
9	16	'parseInt' is not defined.
6	27	'suma' is defined but never used.

Herramienta en línea ideal para encontrar errores de sintaxis, semántica antes de la ejecución de código JavaScript



VALIDAR CODIGO JAVASCRIPT (2)



The screenshot shows the website 'validador de javascript' on the domain 'es.piliapp.com'. The page has a dark header with the URL and a star icon. Below the header, there's a navigation bar with the PiliApp logo and the text 'validador de javascript', along with social media icons for Twitter and Facebook. The main content area features the title 'validador de javascript' in a large, bold font. To the right of the title is a section titled 'Enlaces relacionados' (Related Links) with a list of links: 'Código PHP comprobación de sintaxis', 'MySQL Revisar sintaxis', 'Regex Tester', and 'Comprime JS y CSS'. Below the title is a paragraph describing the tool: 'Javascript Validator es una herramienta de análisis de código estático utilizado en el desarrollo de software para comprobar si el código fuente de JavaScript contiene un error de sintaxis. Ella está compuesta principalmente como una herramienta en línea.' In the center, there is a large text area containing a JavaScript code snippet. At the bottom of this area is the URL 'https://es.piliapp.com/javascript-validator/'. Below the code area is a button labeled 'Compruebe Javascript'.

https://es.piliapp.com/javascript-validator/

Compruebe Javascript

Un validador adicional
encontrar posibles
errores de sintaxis,
semántica antes de la
ejecución de código
JavaScript



VALIDAR CODIGO JAVASCRIPT (3)



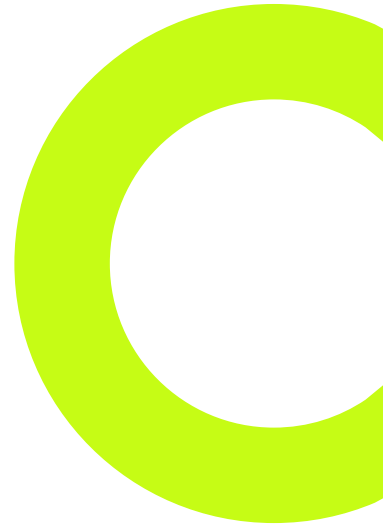
The screenshot shows the website <https://validatejavascript.com> in a browser. The page title is "ValidateJavaScript - Find & Fix JavaScript Errors". The code input area contains the following JavaScript code:

```
1 let ancho, linea, cadena
2 const alto = parseInt(prompt('Teclea un número de 1 a 5'))
3 for (linea = 1; linea <= alto; linea++) {
4   cadena = ''
5   for (ancho = 1; ancho <= linea; ancho++) {
6     cadena += '*'
7   }
8   // console.log( cadena+ "\n");
9   alert(cadena + '\n')
10 }
11
```

Below the code, two error messages are displayed in red boxes:

- 2:23 error 'prompt' is not defined. ([no-undef](#))
- 9:3 error 'alert' is not defined. ([no-undef](#))

At the bottom, there is a button labeled "Find & Fix JavaScript Errors" and a URL bar showing <https://validatejavascript.com/>.



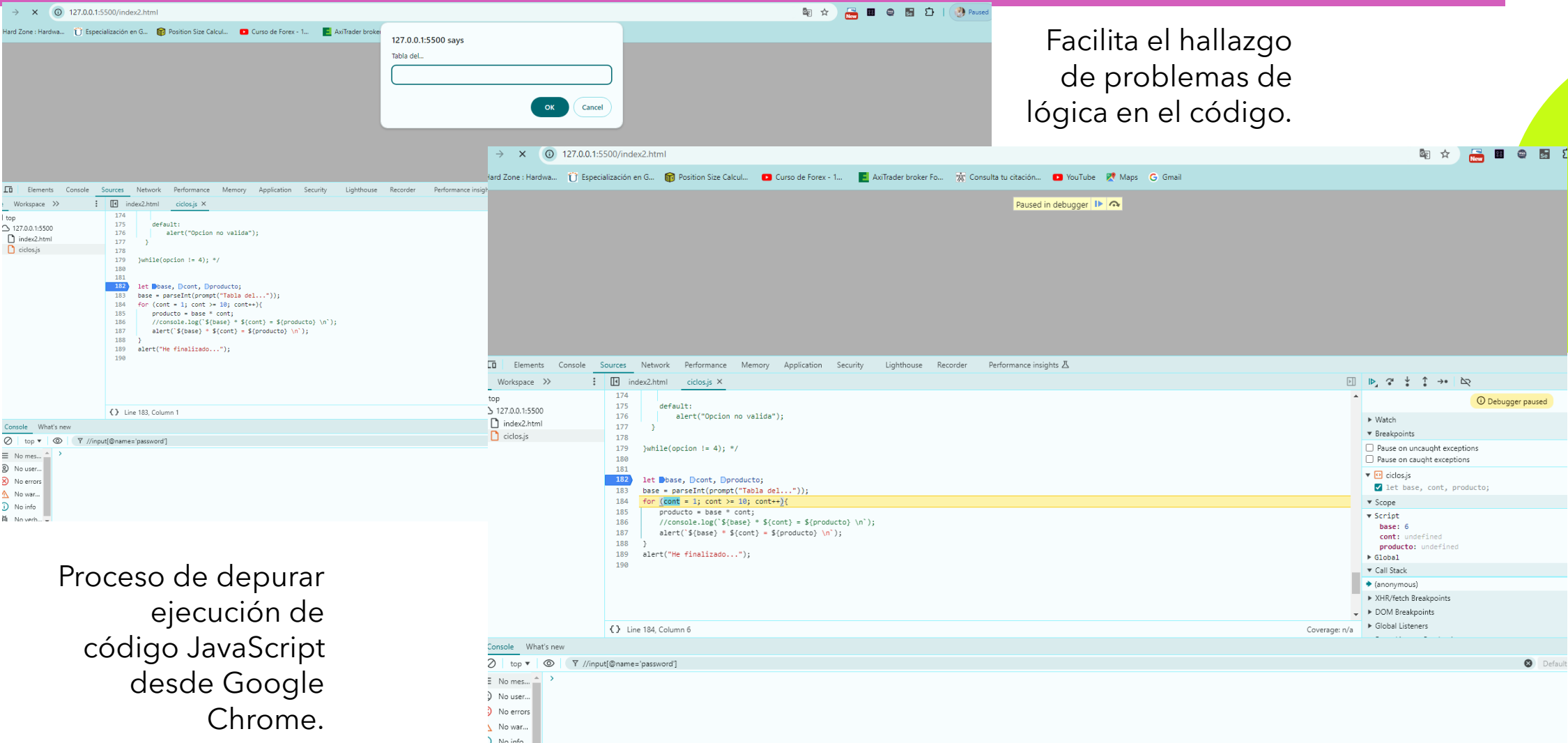
Una más para
encontrar errores de
sintaxis, semántica
antes de la ejecución
de código JavaScript



¿CÓMO REALIZAR DEBUGGER UTILIZANDO NAVEGADOR – GOOGLE CHROME.?

Facilita el hallazgo de problemas de lógica en el código.

Proceso de depurar ejecución de código JavaScript desde Google Chrome.



EJERCICIOS APLICANDO CICLOS REPETITIVOS

- 1- Dibujar un triángulo rectángulo con asteriscos. El usuario tecleará un valor entero, el script escribirá con asteriscos tantas líneas como diga ese número. Cada línea estará formada por una serie de asteriscos tan larga como diga el número de línea en el que está.

Para separar una línea de la siguiente en console o en alert debes usar "\n". En este ejercicio usa console.log.

Ejemplo: Le tecleamos el valor 5. El resultado será:

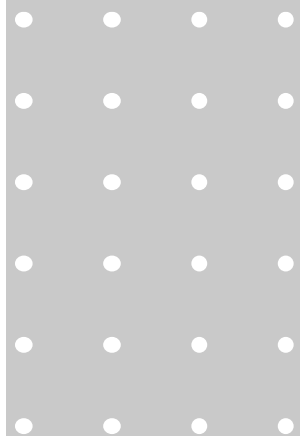
```
*
**
***
****
*****
```

- 2- Crea un script que pida al usuario una contraseña. Si coincide con la clave definida en el script le devolverá el siguiente mensaje "Acceso concedido" de lo contrario le devolverá el mensaje "Acceso Denegado" . Si falla tres veces se visualizará el mensaje "Alerta, intruso tratando de acceder..."

- 3- Implementa un script haciendo uso de "do..while()" para crear una cadena que contenga letras repetidas. Las letras se digitan por teclado y sólo podrán ser la X o la Z, después de elegir las letras se le ingresa el número de repeticiones que deberá estar entre 1 y 15.

Ejemplo: Si elijo X y luego digito 10 se mostrará XXXXXXXXXXXX. Si digito una letra diferente me volverá a pedir la letra, y si tecleo 20 me volverá a pedir el número





PREGUNTAS Y RESPUESTAS