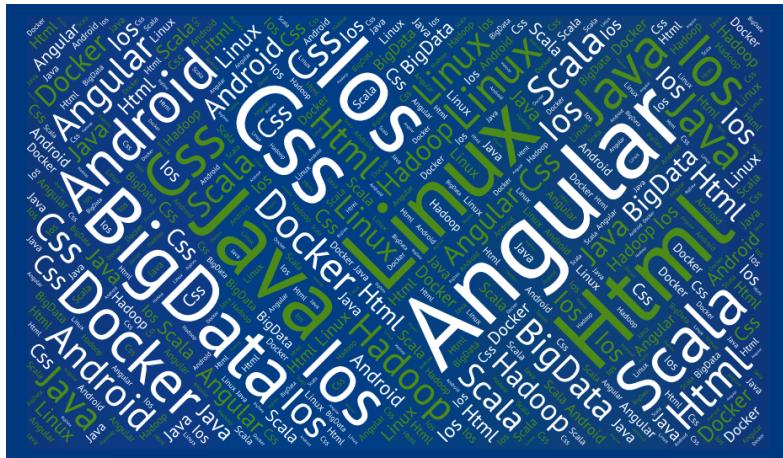




FORMACIÓN EN NUEVAS TECNOLOGÍAS

Angular – TypeScript Avanzado

ICONO TRAINING



Formación en Nuevas Tecnologías



www.iconotc.com

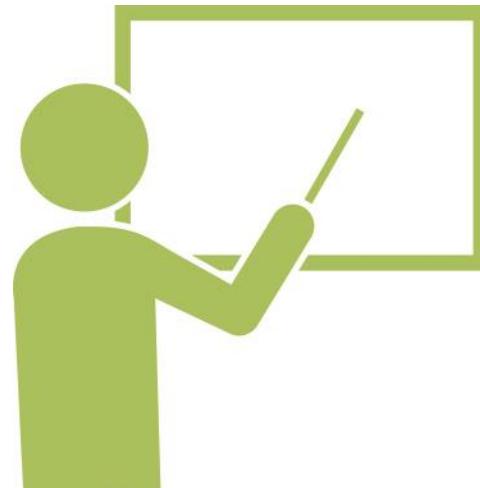


[linkedin.com/company/icono-training-consulting](https://www.linkedin.com/company/icono-training-consulting)



training@iconotc.com

FORMADOR



Rolando López Román



Consultor / formador en Tecnologías de la Información.

¡Síguenos en las Redes Sociales!



ANGULAR – TYPESCRIPT AVANZADO



DURACIÓN

- 25 horas



MODALIDAD

- Remota en tiempo real.



FECHAS y HORARIO:

- Días 21, 22, 23, 24 y 25 Marzo de 2022. De 15:00 hs a 20:30 hs de lunes a jueves y 17:00 hs a 20:00 hs. Viernes.



CONTENIDO:

1. Introducción (Recapitulación puntos clave del curso Iniciación a Angular)
 - a. ECMAScript
 - b. TypeScript
 - c. NVM
 - d. NodeJS
 - e. NPM
 - f. Angular (Arquitectura de proyectos; Configuración de un Proyecto; Dependencias Dev y Prod; Objetos de un proyecto Angular; Componentes; Directivas; Servicios; Pipes)
 - g. Angular CLI (Comandos CLI; Servidor local)
2. Angular avanzado
 - a. Componentes avanzados (Modales, login, spinner, etc)
 - b. Formularios avanzados (validaciones, etc.)
 - c. Routin y Navegación avanzados (Guards)
 - d. Servicios (HttpClient, Observables y RxJS)
 - e. Internacionalización (i18n)
 - f. Test unitarios
 - g. Publicación PROD

Herramientas necesarias

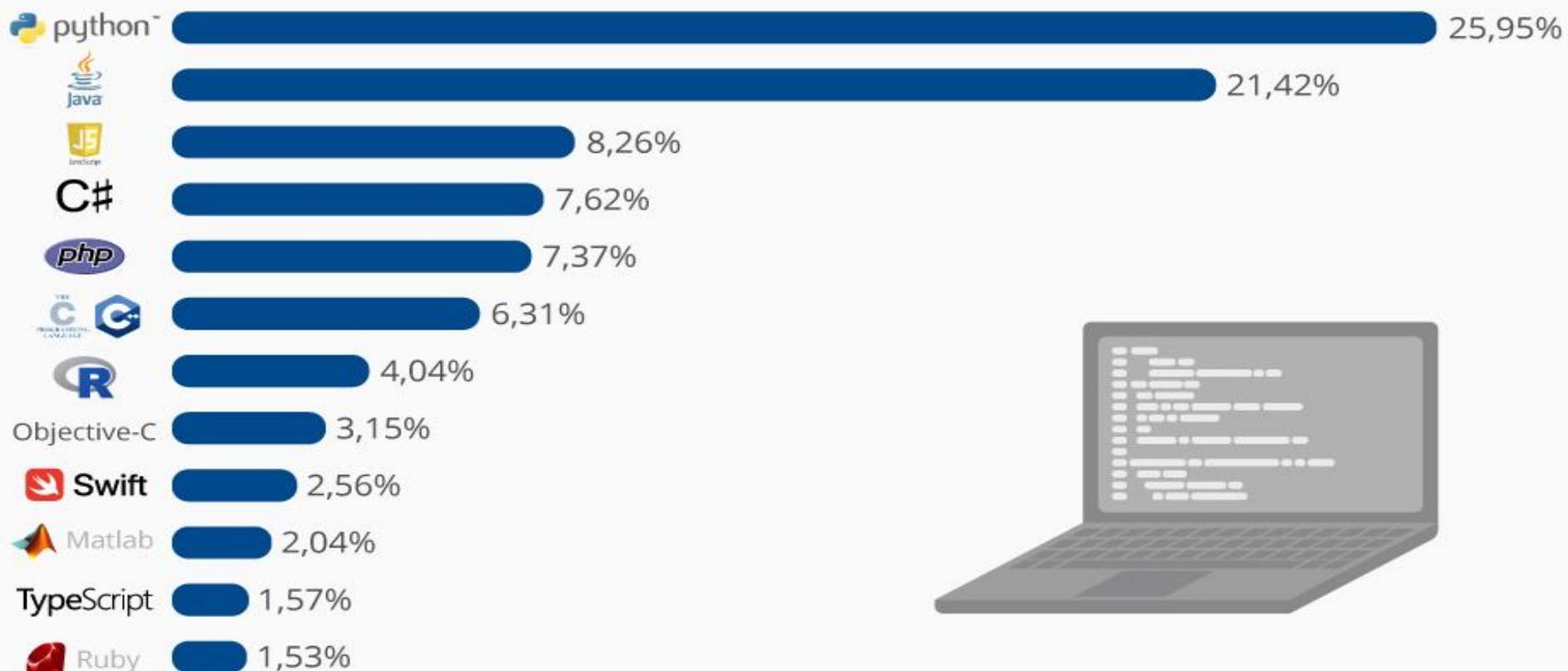
- Visual Studio Code - Editor de código
- Xampp - Servidor Web local

JAVASCRIPT

- ▶ **JavaScript** (abreviado comúnmente **JS**) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.
- ▶ Javascript es un lenguaje poderoso, capaz de aportar soluciones eficaces en la mayoría de los ámbitos de la tecnología.
- ▶ Es especialmente importante porque es el único lenguaje de programación que entienden los navegadores, con el que se desarrolla la parte de la funcionalidad frontend en sitios web y aplicaciones web modernas. Pero también es fundamental en muchos otros tipos de desarrollos. Sus usos más importantes son los siguientes:
- ▶ Desarrollo de sitios web del lado del cliente (frontend, en el navegador)
- ▶ Desarrollo de todo tipo de aplicaciones gracias a la plataforma NodeJS
- ▶ Desarrollo de aplicaciones para dispositivos móviles, híbridas o que compilan a nativo
- ▶ Desarrollo de aplicaciones de escritorio para sistemas Windows, Linux y Mac, pudiendo escribir un código compatible con todas las plataformas.

Los lenguajes de programación más usados

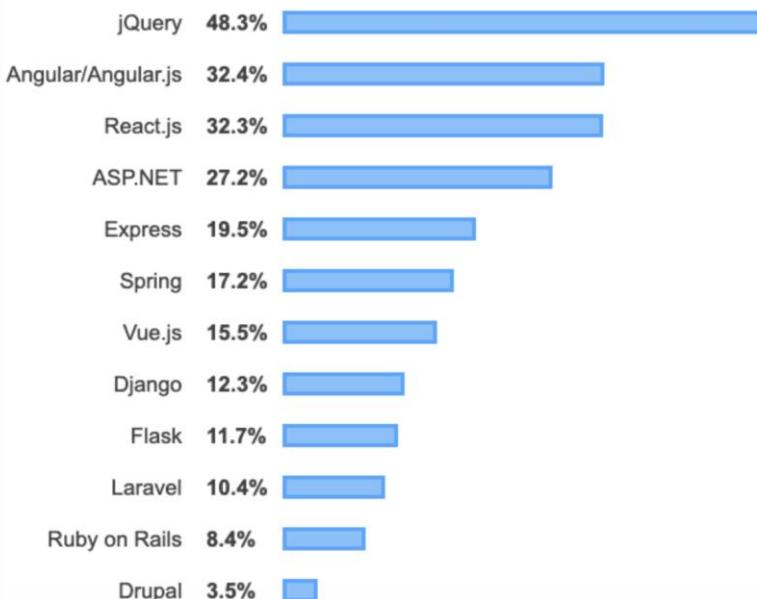
Porcentaje de uso de los lenguajes de programación más populares del mundo*



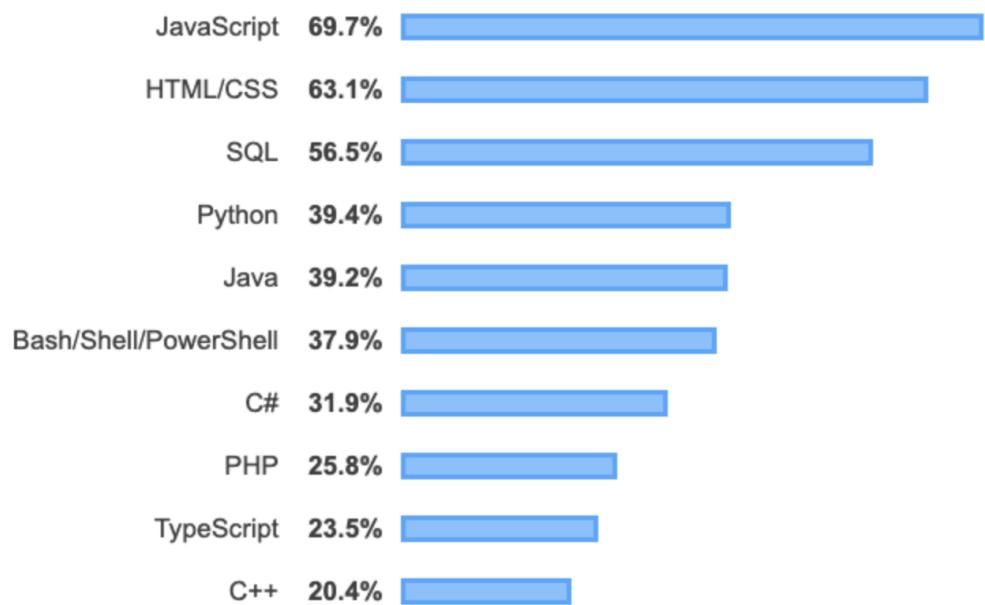
* Datos procedentes del Índice PYPL. Este se basa en las búsquedas en Google de tutoriales de lenguajes de programación.

Datos 2019 - Stack Overflow

Frameworks



Lenguajes de Programación



Versiones de Javascript

- **1996:** LiveScript a JavaScript (estándar)
- **1997:** ES1 (ECMAScript 1)
- **2009:** ES5 (ECMAScript 5) Con muchas características nuevas.
- **2015:** ES6/ES2015 (ECMAScript 2015) que fue la actualización más grande de JavaScript hasta el momento.
- **2015:** Se estableció el año de nuevos lanzamientos de JavaScript.

Cual utilizar?

- ES5:
 - Soportada en todos los navegadores web.
- ES6/ES2015, ES7/ES2016, ES8/ES2017:
 - Soportados por la mayoría de navegadores modernos
 - Pero no por todos los navegadores web
 - Muchas características puedes ser implementadas con polyfills

Pollyfills

Es un código que provee el funcionamiento de una nueva característica de JavaScript (ES6), en versiones viejas como ES5

Entorno de desarrollo

Instalar Node.js para poder ejecutar javascript
desde cualquier parte o dentro del servidor
xammp

Creamos una carpeta del curso/clase01 y dentro un archivo llamado app.js y otro index.html, podemos utilizar el editor Visual Studio Code

The screenshot shows the Visual Studio Code interface with a dark theme. The Explorer sidebar on the left lists a project folder 'CLASE01' containing files 'app.js' and 'index.html'. The 'index.html' file is currently selected and open in the main editor area. The editor displays the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
</body>
</html>
```

The status bar at the bottom right shows the file path 'index.html — clase01', line count 'Ln 10, Col 1', spaces used 'Spaces: 4', encoding 'UTF-8', and file type 'HTML'.

Primer hola mundo

The screenshot shows a developer environment with the following components:

- EXPLORER**: Shows a tree view with **CLASE01** expanded, containing **app.js** and **index.html**.
- OPEN EDITORS**: Shows two tabs: **app.js** and **index.html**. The **index.html** tab is active.
- index.html — clase01**: The content of the **index.html** file is displayed:

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9
10 <script>
11     console.log('Hola Mundo Mba\'eteko');
12 </script>
13
14 </body>
15 </html>
```

A red box highlights the `<script>` block from line 10 to line 12.
- Document**: A browser window showing the URL localhost/curso_javascript.html. The page content is blank.
- Console**: The browser's developer tools Console tab. It shows the output of the highlighted script block:

```
Hola Mundo Mba'eteko
```

A yellow warning message is also present:

⚠ DevTools failed to load SourceMap: Could not load content at http://localhost:3000/curso_javascript.html/build/injectGlobalHook.js.map. HTTP error: status c



<script></script> dentro del html ya no
esta recomendado

The screenshot shows a dark-themed instance of the Visual Studio Code (VS Code) code editor. At the top, the title bar displays "app.js — clase01". The main interface includes the Explorer sidebar on the left, the Command Palette, and the Editor area on the right.

Explorer Sidebar:

- Shows a project structure under "CLASE01":
 - "app.js" (selected, highlighted with a red border)
 - "index.html"
- Other icons include: File, Find, Git, Split View, and Help.

Editor Area:

- The active tab is "app.js".
- The code editor displays the following content:

```
1  console.log('Hola mundo desde app.js');
```

The screenshot shows a developer environment with several windows open:

- EXPLORER**: Shows a tree view of files under "CLASE01". "index.html" is selected.
- OPEN EDITORS**: Shows two editors: "app.js" and "index.html". "index.html" is the active editor.
- index.html — clase01**: The active editor window. It displays the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9
10 <script src="app.js">
11 </script>
12
13 </body>
14 </html>
```

A red box highlights the `<script src="app.js">` line, and a red arrow points from it to the browser's console output.
- Document**: A browser window showing the rendered HTML. The page title is "Document". The URL is "localhost/curso_javascript...". The console output is:

```
⚠ DevTools failed to load SourceMap: Could not load content script
Hasta la vista baby
Hola mundo desde app.js
```
- OUTLINE**: A sidebar showing the outline of the current file.

Sintaxis

- Aunque JavaScript se puede incluir directamente en un documento (X)HTML, ocurre como las hojas de estilo: suele ser más útil como un archivo independiente vinculado a cada documento que necesite de los comportamientos definidos en él; así sólo hay que mantener unos pocos archivos .js para actualizar los comportamiento de todo un sitio¹. Para ello habría que incluir en el head del documento una línea como ésta:
`<script type="text/javascript" src="archivo.js"></script>`
- JavaScript es un lenguaje interpretado, no compilado. Significa que sólo se necesita el bloc de notas para programar un archivo .js, y vincularlo a una página para ver los resultados en un navegador.
- JavaScript es sensible al caso —y muy sensible, además—. Eso quiere decir que, por ejemplo, una variable llamada cadena es completamente distinta a otra llamada Cadena, y que con var matriz = new Array(); se declara una matriz, pero con matriz = new array(); no.
- Aunque no es imprescindible, es una buena práctica que los enunciados terminen en un punto y coma (;). Si éste no existe, el intérprete de JavaScript considerará un salto de línea como el final de un enunciado.
- La sintaxis de los comentarios depende del número de líneas:
`// ...éste es un comentario de una sola linea... /* ...éste es un bloque de comentarios que ocupa varias líneas... */`
Los comentarios, por supuesto, no son interpretados por el agente de usuario.
- Los bloques de código son un conjunto de enunciados que deben interpretarse como una secuencia completa — como por ejemplo una función—. Se indican con llaves ({}).

app.js — clase01

JS app.js X index.html ⋮

JS app.js > ...

```
1 //console.log('Hola mundo desde app.js');
2
3 //comentario en una sola linea javascript
4
5 /*
6 comentarios en lineas en javascript
7 */
8
9 //Variables
10
11 //declaracion de variables
12 let a = 10;
13 //declaración de variables en version antiguas de js
14 var b = 15;
15 //variables constantes no se puede cambiar de valor
16 const c = 20;
17
18 let suma=a+b;
19 console.log('la suma es '+suma);
```

Document localhost/cur...

Elements Console

top

⚠ DevTools failed to load SourceMap: http://build/injectGlobalHook.js.map
la suma es 25

Variables Tipos de datos primitivos

ES UNA INFORMACIÓN QUE NO ES UN OBJETO Y SON INMUTABLES

- Boolean – true/false::Verdadero y Falso
- Null – Sin valor en lo absoluto
- Undefined – Una variable declarada que aún no se le asigna valor
- Number – integers, floats, etc.
- Strings – Una cadena de caracteres, ej: Palabras, nombre de personas
- Symbol – Es un valor único que no es igual a ningún otro valor

primitivos.js X app.js index.html

js > js primitivos.js > ...

```
1 //variables
2 let iva = 16;          // variable tipo entero
3 let total = 234.65;    // variable tipo decimal
4
5 //cadenas de texto
6 let mensaje = "Bienvenido a nuestro sitio web";
7 let nombreProducto = 'Producto ABC';
8 /* El contenido de texto1 tiene comillas simples, por lo que
9 | se encierra con comillas dobles */
10 let texto1 = "Una frase con 'comillas simples' dentro";
11 /* El contenido de texto2 tiene comillas dobles, por lo que
12 | se encierra con comillas simples */
13 let texto2 = 'Una frase con "comillas dobles" dentro';
14 //para dentro de los caracteres
15 //Una nueva linea \n
16 //Un tabulador \t
17 //Una comilla simple \
18 //Una comilla doble \
19 //Una barra inclinada \\
20 let text1 = 'Una frase con \'comillas simples\' dentro';
21 let text2 = "Una frase con \"comillas dobles\" dentro";
22
23 //array
24 let dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"];
25
26 //booleano
27 let clienteRegistrado = false;
28 let ivaIncluido = true;
29 //symbol
30 let symbol1 = Symbol('a');
31 let symbol2 = Symbol('a');
32 console.log(typeof symbol1);
```

Ln 20, Col 57 Spaces: 4 UTF-8 LF JavaScript

Con `typeof` podemos identificar el tipo de variable en `console.log`

Mensajes de consola

The screenshot shows a code editor and a browser developer tools interface. The code editor on the left contains a file named 'app.js' with the following content:

```
app.js — clase01
app.js > ...
52
53
54     let a= 10,
55         b= 20,
56         c= 'Hola',
57         d= 'Estas',
58         x= a+b;
59
60     console.log(a);
61
62     console.log({a});
63     //mensaje de error
64     console.error(b);
65     //mensaje de alerta
66     console.warn(c);
67     //de tipo informativo
68     console.info(d);
69     //vista de tabla con las variables y valores
70     console.table({a,b,c,d,x});
```

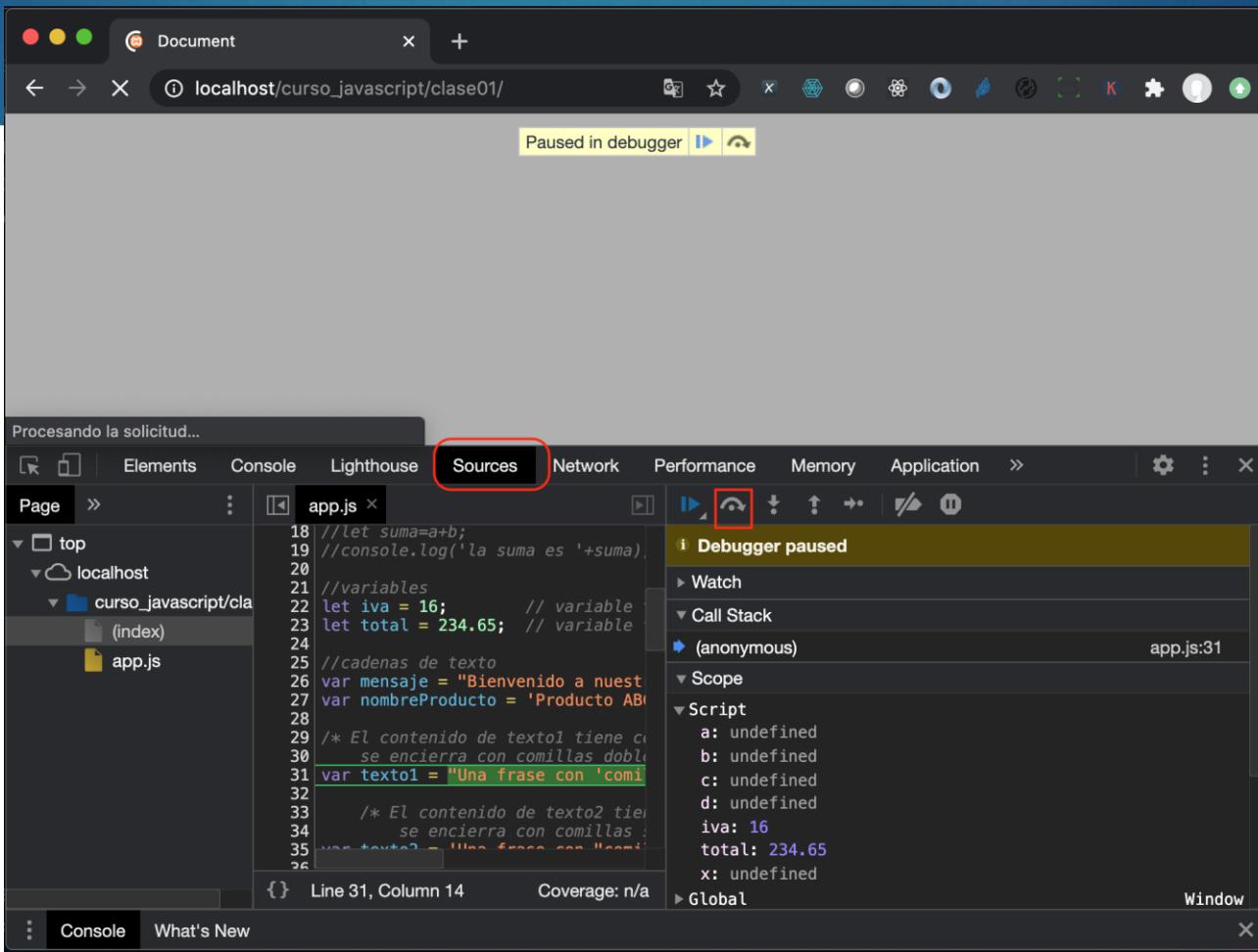
The browser window on the right shows the URL `localhost/curso_javascript/clase01/`. The developer tools console tab is active, displaying the following log output:

- 10
- ▶ {a: 10}
- ✖ ▶ 20
- ⚠ ▶ Hola
- Estas

Below the log, there is a table titled '(index)' showing variable values:

(index)	Value
a	10
b	20
c	"Hola"
d	"Estas"
x	30
▶ Object	

Breakpoints



Creamos esta estructura

The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar: Shows a folder structure under "CLASE01". A red box highlights the "js" folder, which contains "alerts.js", "app.js", and "index.html".
- OPEN EDITORS** tab bar: Shows "alerts.js" and "index.html" (the active tab).
- index.html — clase01**: The active editor tab.
- Content of index.html:**

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9
10 <script src="js/alerts.js"></script>
11
12 </body>
13 </html>
```

A red box highlights the line of code: `<script src="js/alerts.js"></script>`.

Tipos de alertas

The image shows a code editor interface on the left and a web browser window on the right. The code editor has tabs for 'alerts.js' and 'index.html'. The 'alerts.js' tab contains the following code:

```
1
2 alert('Hola esto es una alerta');
```

The browser window shows a single-line alert dialog with the message 'Hola esto es una alerta'. The dialog has a title bar with 'localhost dice' and a blue 'Aceptar' button.

The screenshot shows a developer environment with two tabs: 'alerts.js' and 'index.html'. The 'alerts.js' tab contains the following code:

```
1 //alert('Hola esto es una alerta');
2
3 //nos retorna el dato ingresado
4 //en la ventana del alerta
5 let respuesta = prompt('¿Cual es tu nombre?');
6
7 console.log(respuesta);
```

The 'index.html' tab is currently active. A browser window titled 'Document' is displayed, showing the result of running the JavaScript. The page content is:

localhost dice
¿Cual es tu nombre?
Rolando

At the bottom, the browser's developer tools are visible, showing the 'Console' tab with the following message:

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

The screenshot shows a browser window with developer tools open. The left side displays a code editor with `alerts.js` containing the following JavaScript code:

```
1 //alert('Hola esto es una alerta');
2
3 //nos retorna el dato ingresado
4 //en la ventana del alerta
5 let respuesta = prompt('¿Cual es tu nombre?');
6
7 console.log(respuesta);
```

The right side shows the browser window at `localhost/curso_javascript...`. The developer tools console tab is active, displaying the output of the `console.log` statement. The word "Rolando" is highlighted with a red box.

Console output:

```
⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienih/bundle/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
> Rolando
alerts.js:7
```

The screenshot shows a browser window with two tabs: 'alerts.js' and 'index.html'. The 'alerts.js' tab contains the following code:

```
1 //alert('Hola esto es una alerta');
2
3 //nos retorna el dato ingresado
4 //en la ventana del alerta
5 //let respuesta = prompt('¿Cual es tu nombre?');
6
7 //console.log(respuesta);
8
9 let respuesta=confirm('Estas seguro de que quieres borrar esto');
10 console.log(respuesta);
```

The 'index.html' tab shows a confirmation dialog box with the message "localhost dice" and "Estas seguro de que quieres borrar esto". The "Aceptar" button is highlighted with a red border.

Below the browser, the developer tools are open, showing the Console tab. A warning message is displayed:

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoiienih/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

The screenshot shows a developer setup with two main windows: a code editor and a browser's developer tools.

Code Editor: The left window displays the file `alerts.js` with the following content:

```
1 //alert('Hola esto es una alerta');
2
3 //nos retorna el dato ingresado
4 //en la ventana del alerta
5 //let respuesta = prompt('¿Cuál es tu nombre?');
6 //console.log(respuesta);
7
8
9 let respuesta=confirm('Estás seguro de que quieres borrar esto');
10 console.log(respuesta);
```

Browser DevTools: The right window shows the browser's developer tools with the following details:

- Title Bar:** Document, localhost/curso_javascript...
- Console Tab:** Shows the output of the JavaScript code from `alerts.js`. The output includes:
 - A warning message about failed SourceMap loading: "DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoenihf/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME"
 - The value `true` at line 10 of `alerts.js`.
- Elements, Console, Lighthouse, and other tabs:** Standard DevTools tabs.
- Filter Bar:** Set to "Default levels".

Palabras reservadas

break	export	super
case	extends	switch
catch	finally	this
class	for	throw
const	function	try
continue	if	typeof
debugger	import	var
default	in	void
delete	instanceof	while
do	new	with
else	return	yield
let		
enum	package	public
implements	private	static
interface	protected	await

Evitar usar

null	undefined	true
false	hasOwnProperty	undefined
isNaN	Infinity	isFinite
NaN	length	Math
isPrototypeOf	prototype	valueOf
name	Number	Object
String	toString	prompt
alert	conform	

Arreglos

Son un objeto muy parecido a una lista de información, que contiene un grupo de elementos, usualmente esa información dentro del arreglo es del mismo tipo de dato...

Sintaxis

```
let videoJuegos = ['Mario 3', 'Megaman', 'Chrono Trigger'];
```

0 1 2

```
videoJuegos[1] // Megaman
```

Creamos esta estructura y archivos

The image shows a code editor interface with two main panes. The left pane displays an HTML file named `index.html` with the following content:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    | initial-scale=1.0">
    <title>Arreglos Javascript</title>
</head>
<body>
    <h2>Arreglos Javascript</h2>
</body>
</html>
```

The right pane shows the rendered output in a browser window titled "Arreglos Javascript". The page displays the title "Arreglos Javascript" and the heading "Arreglos Javascript".

No olvidemos de llamar al archivo js en html

The screenshot shows a code editor interface with the following details:

- EXPLORER**: Shows a tree view of files under "CLASE02".
 - A folder named "js" is expanded, showing two files: "arreglos.js" (highlighted with a red underline) and "index.html".
- OPEN EDITORS**: Shows two tabs:
 - arreglos.js**: A JS file containing an array definition.
 - index.html**: An HTML file currently being edited.
- index.html Content (HTML View)**:

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width,
6          initial-scale=1.0">
7      <title>Arreglos Javascript</title>
8  </head>
9  <body>
10     <h2>Arreglos Javascript</h2>
11
12     <script src="js/arreglos.js"></script>
13
14 </body>
15 </html>
```

The line `<script src="js/arreglos.js"></script>` is highlighted with a red rectangular box.

The screenshot shows a code editor and a browser window side-by-side.

Code Editor (Left):

- File tabs: arreglos.js (active), index.html
- Code content:

```
js > js arreglos.js > ...
1 //formas de declarar los arreglos
2
3 //const arreglo = new Array(10);
4
5 //const arreglo = [];
6
7
8 //console.log(arreglo);
9
10 let videoJuegos=['Mario','Sonic','Megaman','Contra'];
11
12
13 console.log({videoJuegos});
14 console.log(videoJuegos[0]);
15
```

Browser (Right):

- Title bar: Arreglos Javascript
- Address bar: localhost/curso_javascript...
- Section title: Arreglos Javascript
- Console tab (DevTools):

 - Elements, Console, Lighthouse, etc. buttons
 - Top frame: top
 - Warning message: DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkienihf/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
 - Object inspection: {videoJuegos: Array(4)}
 - videoJuegos: (4) ["Mario", "Sonic", "Megaman", "Contra"]
 - __proto__: Object
 - Mario
 - arreglos.js:14

- Bottom navigation: Console, What's New, etc.

JS arreglos.js X index.html

```
js > JS arreglos.js > ...
8     //console.log(arreglo);
9
10    //let videoJuegos=['Mario','Sonic','Megaman','Cont
11
12
13    //console.log({videoJuegos});
14    //console.log(videoJuegos[0]);
15
16    let arregloCosas = [
17        true,
18        123,
19        'Fernando',
20        1+2,
21        function(){}
22        ()=>{},
23        {a: 1},
24        ['Donkey kong','Macross','Mario Bros 3']
25
26    ];
27
28    console.log({arregloCosas});
29    console.log(arregloCosas[7][0]);
```

Arreglos Javascript

localhost/curso_javascript...

Arreglos Javascript

Elements Console Lighthouse >

top Filter Default levels

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkienihf/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

▶ {arregloCosas: Array(8)} arreglos.js:28

▶ Donkey kong arreglos.js:29

⋮

Console What's New



✖ 0 ⚡ 0

Ln 29, Col 33

Spaces: 4

UTF-8 LF

JavaScript



Ahora veremos algunas funciones útiles para arreglos

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left lists files in the 'CLASE02' folder, specifically 'js/arreglos.js' and 'js/arreglos2.js'. The 'arreglos2.js' file is currently selected and highlighted with a red box. The Editor pane on the right displays the 'index.html' file, which contains the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Arreglos Javascript</title>
</head>
<body>
    <h2>Arreglos Javascript</h2>
    <!--<script src="js/arreglos.js"></script>
    <script src="js/arreglos2.js"></script>
</body>
</html>
```

The line containing the second script tag, 'arreglos2.js', is also highlighted with a red box.

The screenshot shows a browser developer tools interface with the "Console" tab selected. The code being run is:

```
js > js arreglos2.js > juegos.forEach() callback > arr  
1  
2  
3 let juegos = ['Zelda','Mario Kart','Metroid','Tetris'];  
4  
5 //length para saber cantidad de elementos  
6 // que contiene nuestro arreglo  
7  
8 //console.log(juegos.length);  
9  
10 //foreach para recorrer nuestros elementos  
11  
12 juegos.forEach((elemento,indice,arr) => {  
13     console.log([elemento,indice,arr]);  
14 } );
```

The output in the console is:

```
arreglos Javascript  
Arreglos Javascript  
Elements Console  
top  
Filter Default  
⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoenihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME  
arreglos2.js:14 {elemento: "Zelda", indice: 0, arr: Array(4)}  
arreglos2.js:14 ► arr: (4) ["Zelda", "Mario Kart", "Metroid", ...]  
arreglos2.js:14 ► elemento: "Zelda"  
arreglos2.js:14 ► indice: 0  
arreglos2.js:14 ► __proto__: Object  
arreglos2.js:14 {elemento: "Mario Kart", indice: 1, arr: Array(4)}  
arreglos2.js:14 {elemento: "Metroid", indice: 2, arr: Array(4)}  
arreglos2.js:14 {elemento: "Tetris", indice: 3, arr: Array(4)}  
Console What's New
```

```
2
3     let juegos = ['Zelda', 'Mario Kart',
4     'Metroid', 'Tetris'];
5
6     //length para saber cantidad de elementos
7     // que contiene nuestro arreglo
8
9     //console.log(juegos.length);
10
11    //foreach para recorrer nuestros elementos
12
13    /*juegos.forEach((elemento,indice,arr) => {
14
15        console.log({elemento,indice,arr});
16
17   });*/
18
19    //push agrega un elemento al final del arreglo
20    let nuevoJuego = juegos.push('Ice Climber');
21    //console.log({nuevoJuego, juegos});
22    //unshift agrega un elemento al principio
23    // del arreglo
24    nuevoJuego = juegos.unshift('F-Zero');
25    console.log({nuevoJuego, juegos});
```

Ln 4, Col 21 Spaces: 4 UTF-8 LF JavaScript ✎

Arreglos Javascript

Elements Console Lighthouse Sources Network >

top Filter Default levels

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fm_kadmapgofadopljbjfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

arreglos2.js:25

▼ {nuevoJuego: 6, juegos: Array(6)}
 ▶ juegos: (6) ["F-Zero", "Zelda", "Mario Kart", "Metroid", "Tetris", "Ice Climber"]
 nuevoJuego: 6
 ▶ __proto__: Object

Console What's New

The image shows a developer environment with two main panes. On the left is a code editor with a dark theme, displaying a JavaScript file named 'arreglos2.js'. The code uses array methods like push, unshift, and pop to manipulate an array of video game titles. On the right is a browser window titled 'Arreglos Javascript' showing the output of the code. The browser's developer tools are open, specifically the 'Console' tab, which displays the final state of the 'juegos' array and the values of the variables 'nuevoJuego' and 'borrarJuego'. A warning message in the console indicates that a SourceMap could not be loaded due to an HTTP error (status code 404).

```
js arreglos2.js X index.html
js arreglos2.js > ...
15
16  });
17 let juegos = ['Zelda', 'Mario Kart',
18 'Metroid', 'Tetris'];
19
20 //push agrega un elemento al final del arreglo
21 let nuevoJuego = juegos.push('Ice Climber');
22 //console.log({nuevoJuego, juegos});
23 //unshift agrega un elemento al principio
24 // del arreglo
25 nuevoJuego = juegos.unshift('F-Zero');
26 console.log({nuevoJuego, juegos});
27
28 //pop borra el ultimo elemento del array
29 let borrarJuego = juegos.pop();
30 console.log({borrarJuego, juegos});
```

Arreglos Javascript

```
Elements Console Lighthouse Sources Network > Default levels ▾
top Filter
⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoiienihf/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
▶ {nuevoJuego: 6, juegos: Array(6)} arreglos2.js:26
▶ {borrarJuego: "Ice Climber", juegos: Array(5)} arreglos2.js:30
  ▶ borrarJuego: "Ice Climber"
  ▶ juegos: (5) ["F-Zero", "Zelda", "Mario Kart", "Metroid", "Tetris"]
  ▶ __proto__: Object
>
```

The image shows a code editor and a browser developer tools panel side-by-side.

Code Editor (Left):

```
js > js arreglos2.js > ...
16  });
17  let juegos = ['Zelda','Mario Kart',
18  'Metroid','Tetris'];
19
20 //push agrega un elemento al final del arreglo
21 let nuevoJuego = juegos.push('Ice Climber');
22 //console.log({nuevoJuego, juegos});
23 //unshift agrega un elemento al principio
24 // del arreglo
25 nuevoJuego = juegos.unshift('F-Zero');
26 console.log({nuevoJuego, juegos});
27
28 //pop borra el ultimo elemento del array
29 let borrarJuego = juegos.pop();
30 console.log({borrarJuego, juegos});
31
32 //borrar elementos especificos del array
33
34 let pos=1;
35 let juegosBorrados = juegos.splice(pos,2);
36 console.log({juegosBorrados,juegos});
```

A red box highlights the last three lines of code (lines 34-36).

Browser Developer Tools (Right):

The browser title is "Arreglos Javascript" and the URL is "localhost/curso_java...".

The developer tools panel shows the following object structure:

- {nuevoJuego: 6, juegos: Array(6)} arreglos2.js:26
- {borrarJuego: "Ice Climber", juegos: Array(5)} arreglos2.js:30
- {juegosBorrados: Array(2), juegos: Array(3)} arreglos2.js:36
 - juegos: (3) ["F-Zero", "Metroid", "Tetris"]
 - juegosBorrados: (2) ["Zelda", "Mario Kart"]
 - __proto__: Object

An arrow points from the highlighted code in the editor to the "juegosBorrados" entry in the developer tools panel.

The screenshot shows a browser developer tools interface with two tabs: "Arreglos Javascript" and "Console".

Arreglos Javascript Tab:

- URL: localhost/curso_java...
- Title: Arreglos Javascript
- Content: "Arreglos Javascript"

Console Tab:

- Elements, Console, Lighthouse, Sources, Network, etc. tabs are visible.
- Filter bar: top, Default levels.
- Message: DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
- Object inspection tree:
 - {nuevoJuego: 6, juegos: Array(6)}
 - {borrarJuego: "Ice Climber", juegos: Array(5)}
 - {juegosBorrados: Array(2), juegos: Array(3)}
 - {posicionJuego: 2, juegos: Array(3)}
 - juegos: (3) ["F-Zero", "Metroid", "Tetris"]
 - posicionJuego: 2
 - __proto__: Object

Code Editor (arreglos2.js):

```
js > JS arreglos2.js > ...
17 let juegos = ['Zelda', 'Mario Kart',
18 'Metroid', 'Tetris'];
19
20 //push agrega un elemento al final del arreglo
21 let nuevoJuego = juegos.push('Ice Climber');
22 //console.log({nuevoJuego, juegos});
23 //unshift agrega un elemento al principio
24 // del arreglo
25 nuevoJuego = juegos.unshift('F-Zero');
26 console.log({nuevoJuego, juegos});
27
28 //pop borra el ultimo elemento del array
29 let borrarJuego = juegos.pop();
30 console.log({borrarJuego, juegos});
31
32 //borrar elementos especificos del array
33
34 let pos=1;
35 let juegosBorrados = juegos.splice(pos,2);
36 console.log({juegosBorrados,juegos});
37
38 //encontrar posicion de un array
39 let posicionJuego = juegos.indexOf('Tetris');
40 console.log({posicionJuego,juegos});
```

A red box highlights the code from line 38 to 40, and a red arrow points from this highlighted area to the "posicionJuego" entry in the inspection tree.

Arreglos, objetos literales

The screenshot shows a portion of the Visual Studio Code interface. On the left is the Explorer sidebar with a tree view of files. In the center is the main editor area displaying the content of `index.html`. The file contains HTML code with a script tag pointing to `arreglos-literales.js`, which is highlighted with a red oval. The `index.html` tab is active at the top.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>Arreglos Javascript</title>
</head>
<body>
    <h2>Arreglos Javascript</h2>
    <!--<script src="js/arreglos.js"></script>-->
    <script src="js/arreglos-literales.js"></script>
</body>
</html>
```

js arreglos-literales.js ×

```
js > js arreglos-literales.js > ...
1
2  let personaje = {
3      nombre: 'Tony Stark',
4      nombreClave: 'Iron Man',
5      vivo: false,
6      edad: 40,
7      coordenadas: {
8          lat: 0.3234343,
9          lng: -23.2343534
10     },
11     trajes: ['Mark I', 'Mark XXV', 'Hulkbuster'],
12     direccion: {
13         zip: '10880 , 90265',
14         ubicacion: 'Malibu, California'
15     }
16 };
17
18 };
19
20 console.log(personaje);
21
22 console.log('Nombre', personaje.nombre);
```

Arreglos Javascript

localhost/curso_javascript...

Arreglos Javascript

Elements Console Lighthouse Default levels

top

arreglos-literales.js:20

{nombre: "Tony Stark", nombreClave: "Iron Man", vivo: false, edad: 40, coordenadas: {...}, ...}

coordenadas:

lat: 0.3234343
lng: -23.2343534
__proto__: Object

direccion:

ubicacion: "Malibu, California"
zip: "10880 , 90265"
__proto__: Object

edad: 40
nombre: "Tony Stark"
nombreClave: "Iron Man"

trajes: Array(3)
0: "Mark I"
1: "Mark XXV"
2: "Hulkbuster"
length: 3
__proto__: Array(0)

vivo: false
__proto__: Object

Nombre Tony Stark

arreglos-literales.js:22

Console What's New

The image shows a code editor on the left and a browser developer tools console on the right. The code editor displays a JavaScript file named 'arreglos-literales.js' with the following content:

```
1 let personaje = {  
2     nombre: 'Tony Stark',  
3     nombreClave: 'Iron Man',  
4     vivo: false,  
5     edad: 40,  
6     coordenadas: {  
7         lat: 0.3234343,  
8         lng: -23.2343534  
9     },  
10    trajes: ['Mark I', 'Mark XXV', 'Hulkbuster'],  
11    direccion: {  
12        zip: '10880 , 90265',  
13        ubicacion: 'Malibu, California'  
14    }  
15};  
16  
17 console.log(personaje);  
18  
19 console.log('Nombre', personaje.nombre);  
20 console.log('Edad', personaje['edad']);  
21  
22 console.log('Cantidad de trajes:', personaje.trajes.length);  
23  
24 console.log('El ultimo traje:',  
25 personaje.trajes[personaje.trajes.length - 1]);  
26  
27 let x='vivo';  
28 console.log('Vivo', personaje[x]);
```

A red box highlights the code from line 19 to line 28. An arrow points from this highlighted area to the browser's developer tools console on the right.

The browser developer tools console shows the output of the code:

- Line 20: {nombre: "Tony Stark", nombreClave: "Iron Man", vivo: false, edad: 40, coordenadas: {...}, ...}
- Line 22: Nombre Tony Stark
- Line 23: Edad 40
- Line 25: Cantidad de trajes: 3
- Line 27: El ultimo traje: Hulkbuster
- Line 31: Vivo false

The browser title bar says 'Arreglos Javascript' and the address bar shows 'localhost/curso_javascript...'. The developer tools tabs include Elements, Console, Lighthouse, and Default levels.

Funciones y características de los objetos

The screenshot shows a browser developer tools window with two main tabs: "Elements" and "Console".

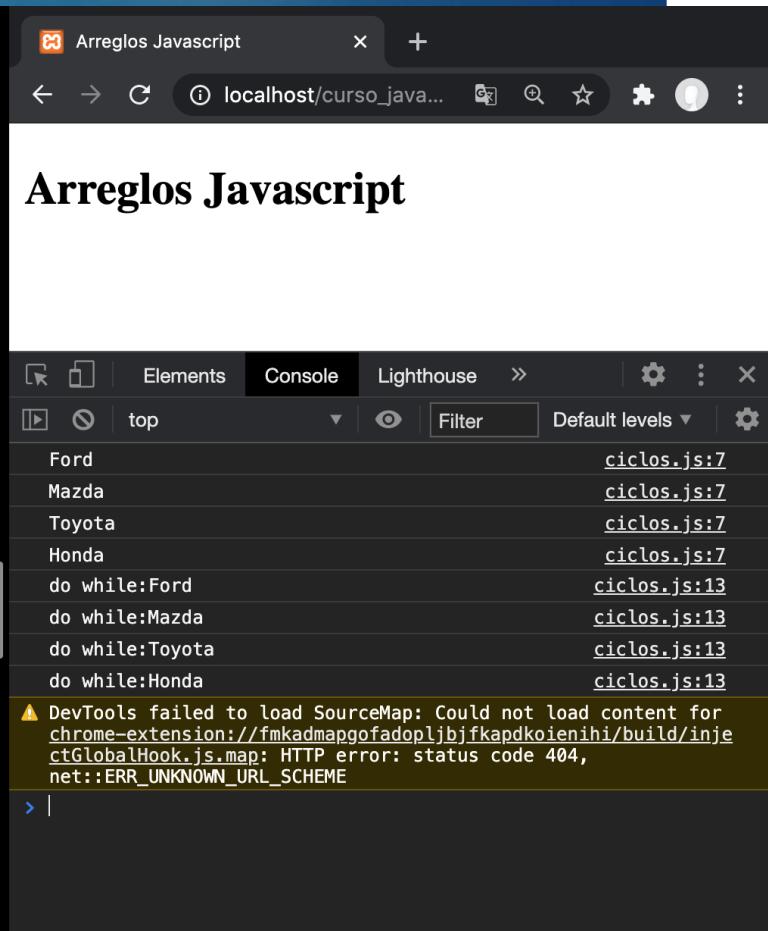
Elements Tab: Displays the structure of an object named "personaje". The object has properties: "vivo" (value: "vivo"), "cordenadas" (value: "cordenadas"), "trajes" (value: "trajes"), "direccion" (value: "direccion"), and "casado" (value: "casado"). It also has a "length" property (value: 7) and two prototype properties: "__proto__: Array(0)" and "__proto__: Object".

Console Tab: Shows the output of several JavaScript console.log statements from a file named "arreglos-literales.js". The logs include:

```
js > js arreglos-literales.js > ...
30 let x='vivo';
31 console.log('Vivo',personaje[x]);
32
33
34 //borrar un objeto específico del objeto
35 delete personaje.edad;
36 console.log(personaje);
37
38 //agregar objetos
39 personaje.casado = true;
40 console.log('Casado',personaje.casado);
41
42 //vista de objetos indice y valor
43
44 const entriesPares = Object.entries(personaje);
45 console.log(entriesPares);
46
47
48 //vista de array
49 const propiedades = Object.getOwnPropertyNames(personaje);
50 console.log({propiedades});
51 //vista de array y valores
52 const propiedades2 = Object.values(personaje);
53 console.log({propiedades2,propiedades2});
```

At the bottom of the developer tools, there is a status bar with the following information: "Ln 54, Col 1 Spaces: 4 UTF-8 LF JavaScript".

CICLOS



The screenshot shows a browser window titled "Arreglos Javascript" at "localhost/curso_java...". The page content is "Arreglos Javascript". Below the content, the developer tools' "Console" tab is open, displaying the following log entries:

Message	Source
Ford	ciclos.js:7
Mazda	ciclos.js:7
Toyota	ciclos.js:7
Honda	ciclos.js:7
do while:Ford	ciclos.js:13
do while:Mazda	ciclos.js:13
do while:Toyota	ciclos.js:13
do while:Honda	ciclos.js:13

A yellow warning message at the bottom of the console states: "⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienihf/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME".

On the left side of the image, there is a code editor window for "ciclos.js" showing the following code:

```
1 let autos=['Ford','Mazda','Toyota','Honda'];
2
3 let i=0;
4
5 while(i<autos.length){
6     console.log(autos[i]);
7     i++;
8 }
9
10 let j=0;
11 do{
12     console.log('do while:' + autos[j]);
13     j++;
14 }while(j < autos.length);
15
16
17
```

ciclos.js index.html

```
js > js ciclos.js > ...
1
2  let hero=['Superman','Batman','LinternaVerde','Flash'];
3
4  for (let i = 0; i < hero.length; i++) {
5    console.log(hero[i]);
6
7  }
8
9  for(let i in hero){
10    console.log('Segunda opción:' +hero[i]);
11  }
12
13  for(let heros of hero){
14    console.log(`Tercera opción ${heros}`);
15  }
16
17
18
19  /*let autos=['Ford','Mazda','Toyota','Honda'];
20
21  let i=0;
22
23  while(i<autos.length){
24    console.log(autos[i]);
25    i++;
26  }
```

...

Arreglos Javascript

localhost/curso_javascript

Arreglos Javascript

Elements Console Lighthouse Default levels

top Filter

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

Output	File
Superman	ciclos.js:5
Batman	ciclos.js:5
LinternaVerde	ciclos.js:5
Flash	ciclos.js:5
Segunda opción:Superman	ciclos.js:10
Segunda opción:Batman	ciclos.js:10
Segunda opción:LinternaVerde	ciclos.js:10
Segunda opción:Flash	ciclos.js:10
Tercera opción:Superman	ciclos.js:14
Tercera opción:Batman	ciclos.js:14
Tercera opción:LinternaVerde	ciclos.js:14
Tercera opción:Flash	ciclos.js:14

Funciones

The screenshot shows the Visual Studio Code interface. The title bar displays "Funciones". The left sidebar has icons for file operations like Open, Save, Find, and Refresh. The Explorer sidebar shows a project structure under "CLASE02": "js" folder containing "arreglos-literal...", "arreglos.js", "arreglos2.js", and "funciones.js" (which is highlighted with a red box). The main editor area has two tabs: "funciones.js" (active) and "index.html". The "funciones.js" tab contains a single line of code: `function saludar(nombre){ console.log("Hola "+nombre); }`. The "index.html" tab shows the following code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Arreglos Javascript</title>
7 </head>
8 <body>
9   <h2>Arreglos Javascript</h2>
10
11
12  <!--<script src="js/arreglos.js"></script>-->
13  <script src="js/funciones.js"></script>
14
15 </body>
16 </html>
```

The line `<script src="js/funciones.js"></script>` in the "index.html" code is also highlighted with a red box.

The screenshot shows a browser window with the title "Arreglos Javascript" and the URL "localhost/curso_javas...". The developer tools console tab is selected, showing the following log entries:

```
Hola 1
Hola 2
Hola 3
```

Each log entry is preceded by its source file and line number: "funciones.js:4", "funciones.js:10", and "funciones.js:14". Above the logs, there is a warning message from DevTools:

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoinihf/build/injectGlobalHok.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

The screenshot shows a browser window with the title "Arreglos Javascript" and the URL "localhost/curso_javas...". The developer tools console tab is selected, showing the following output:

```
Hola 1 Rolando
Hola 2 Jorge
Hola 3 Gustavo
```

Below the console, a message in the DevTools panel states: "DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoinihf/build/injectGlobalHok.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME".

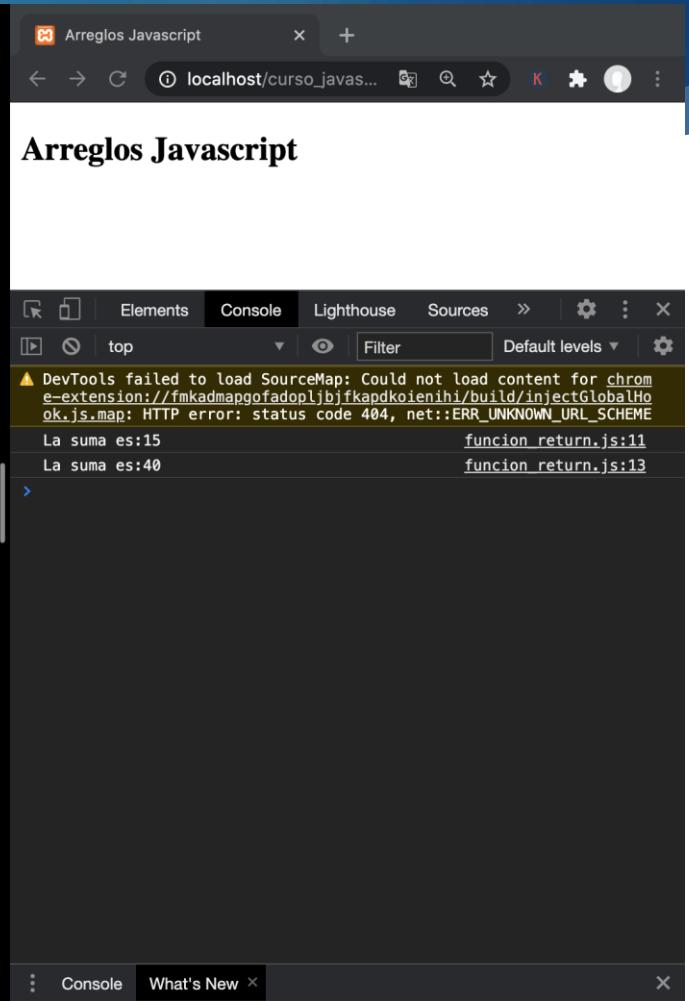
On the left, there is a sidebar with various icons, and at the bottom, there are status bars for "Ln 19, Col 18", "Spaces: 4", "UTF-8", "LF", "JavaScript", and "Console" and "What's New" tabs.

Return

The screenshot shows a code editor interface with the following details:

- File Tabs:** The top bar shows two tabs: "funcion_return.js X" (active) and "index.html".
- Sidebar Icons:** On the left side, there are several icons: a document icon, a magnifying glass, a circular arrow, a square with a circle, a person icon, and a gear icon.
- Code Area:** The main area displays the following JavaScript code:

```
1
2  ↘ function sumar(a,b){
3      |     return a+b;
4  }
5
6
7  ↘ const sumar2 = (a,b) =>{
8      |     return a+b;
9  }
10
11 console.log('La suma es:'+sumar(10,5));
12
13 console.log('La suma es:'+sumar2(25,15));|
```
- Status Bar:** At the bottom, the status bar indicates: "Ln 13, Col 42", "Spaces: 4", "UTF-8", "LF", "JavaScript", and a set of small navigation icons.



Operadores de Asignación

Nombre	Operador abreviado	Significado
Asignación	<code>x = y</code>	<code>x = y</code>
Asignación de adición	<code>x += y</code>	<code>x = x + y</code>
Asignación de resta	<code>x -= y</code>	<code>x = x - y</code>
Asignación de multiplicación	<code>x *= y</code>	<code>x = x * y</code>
Asignación de división	<code>x /= y</code>	<code>x = x / y</code>
Asignación de residuo	<code>x %= y</code>	<code>x = x % y</code>
Asignación de exponenciación	<code>x **= y</code>	<code>x = x ** y</code>
Asignación de desplazamiento a la izquierda	<code>x <= y</code>	<code>x = x << y</code>
Asignación de desplazamiento a la derecha	<code>x >= y</code>	<code>x = x >> y</code>
Asignación de desplazamiento a la derecha sin signo	<code>x >>= y</code>	<code>x = x >>> y</code>
Asignación AND bit a bit	<code>x &= y</code>	<code>x = x & y</code>
Asignación XOR bit a bit	<code>x ^= y</code>	<code>x = x ^ y</code>
Asignación OR bit a bit	<code>x = y</code>	<code>x = x y</code>
Asignación AND lógico	<code>x &&= y</code>	<code>x && (x = y)</code>
Asignación OR lógico	<code>x = y</code>	<code>x (x = y)</code>
Asignación de anulación lógica	<code>x ??= y</code>	<code>x ?? (x = y)</code>

Operador de Comparación

Operador	Descripción	Ejemplos que devuelven true
Igual (==)	Devuelve <code>true</code> si los operandos son iguales.	<code>3 == var1</code> <code>"3" == var1</code> <code>3 == '3'</code>
No es igual (!=)	Devuelve <code>true</code> si los operandos <i>no</i> son iguales.	<code>var1 != 4</code> <code>var2 != "3"</code>
Estrictamente igual (===)	Devuelve <code>true</code> si los operandos son iguales y del mismo tipo. Consulta también Object.is y similitud en JS .	<code>3 === var1</code>
Desigualdad estricta (!==)	Devuelve <code>true</code> si los operandos son del mismo tipo pero no iguales, o son de diferente tipo.	<code>var1 !== "3"</code> <code>3 !== '3'</code>
Mayor que (>)	Devuelve <code>true</code> si el operando izquierdo es mayor que el operando derecho.	<code>var2 > var1</code> <code>"12" > 2</code>
Mayor o igual que (>=)	Devuelve <code>true</code> si el operando izquierdo es mayor o igual que el operando derecho.	<code>var2 >= var1</code> <code>var1 >= 3</code>
Menor que (<)	Devuelve <code>true</code> si el operando izquierdo es menor que el operando derecho.	<code>var1 < var2</code> <code>"2" < 12</code>
Menor o igual (<=)	Devuelve <code>true</code> si el operando izquierdo es menor o igual que el operando derecho.	<code>var1 <= var2</code> <code>var2 <= 5</code>

Operadores Aritmeticos

Operador	Descripción	Ejemplo
Residuo (%)	Operador binario. Devuelve el resto entero de dividir los dos operandos.	12 % 5 devuelve 2.
Incremento (++)	Operador unario. Agrega uno a su operando. Si se usa como operador prefijo (++x), devuelve el valor de su operando después de agregar uno; si se usa como operador sufijo (x++), devuelve el valor de su operando antes de agregar uno.	Si <code>x</code> es 3, <code>++x</code> establece <code>x</code> en 4 y devuelve 4, mientras que <code>x++</code> devuelve 3 y, solo entonces, establece <code>x</code> en 4.
Decremento (--)	Operador unario. Resta uno de su operando. El valor de retorno es análogo al del operador de incremento.	Si <code>x</code> es 3, entonces <code>--x</code> establece <code>x</code> en 2 y devuelve 2, mientras que <code>x--</code> devuelve 3 y, solo entonces, establece <code>x</code> en 2.
Negación unaria (-)	Operador unario. Devuelve la negación de su operando.	Si <code>x</code> es 3, entonces <code>-x</code> devuelve -3.
Positivo unario (+)	Operador unario. Intenta convertir el operando en un número, si aún no lo es.	<code>+"3"</code> devuelve 3. <code>+true</code> devuelve 1.
Operador de exponentiación (**)	Calcula la <code>base</code> a la potencia de <code>exponente</code> , es decir, <code>base^{exponente}</code>	<code>2 ** 3</code> returns 8. <code>10 ** -1</code> returns 0.1.

IF ELSE

A screenshot of a code editor window. The title bar shows two tabs: "index.html" and "if-else.js". The "if-else.js" tab is active. Below the tabs, a breadcrumb navigation bar indicates the current file path: "js > if-else.js > ...". The main editor area contains the following JavaScript code:

```
1
2 let a=10;
3
4
5 if(a==10){
6     console.log('a es mayor a 10');
7 }else{
8     console.log('fin del programa');
9 }
10
11
12
13
14
15
16
```

A screenshot of a browser developer tools window, specifically the "Console" tab. The title bar shows the URL "localhost/curso_javascript...". The main content area displays the text "Arreglos Javascript". Below the content, the developer tools interface includes tabs for "Elements", "Console", "Lighthouse", and "Sources". The "Console" tab is active. The console log output is as follows:

```
DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienihi/build/injectGlobalHook.js.map:
HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
a es mayor a 10
if-else.js:6
```

The screenshot shows a browser window with the title "Arreglos Javascript" at the URL "localhost/curso_javascript...". The browser's developer tools are open, specifically the Console tab. A warning message is displayed: "⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoiienih/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME". Below the warning, the output of a JavaScript log statement is shown: "a es menor a 10" followed by the file name "if-else.js:18".

index.html js if-else.js X

js > js if-else.js > ...

```
1
2     let a=8;
3
4
5
6     /*if(a=10){
7         console.log('a es mayor a 10');
8     }else{
9         console.log('fin del programa');
10    }*/
11
12
13     if(a == 10){
14         console.log('a es igual a 10');
15     }else if(a > 10){
16         console.log('a es mayor a 10');
17     }else if(a < 10){
18         console.log('a es menor a 10');
19     }
20
21
22
23
```

Arreglos Javascript

Elements Console Lighthouse Sources >

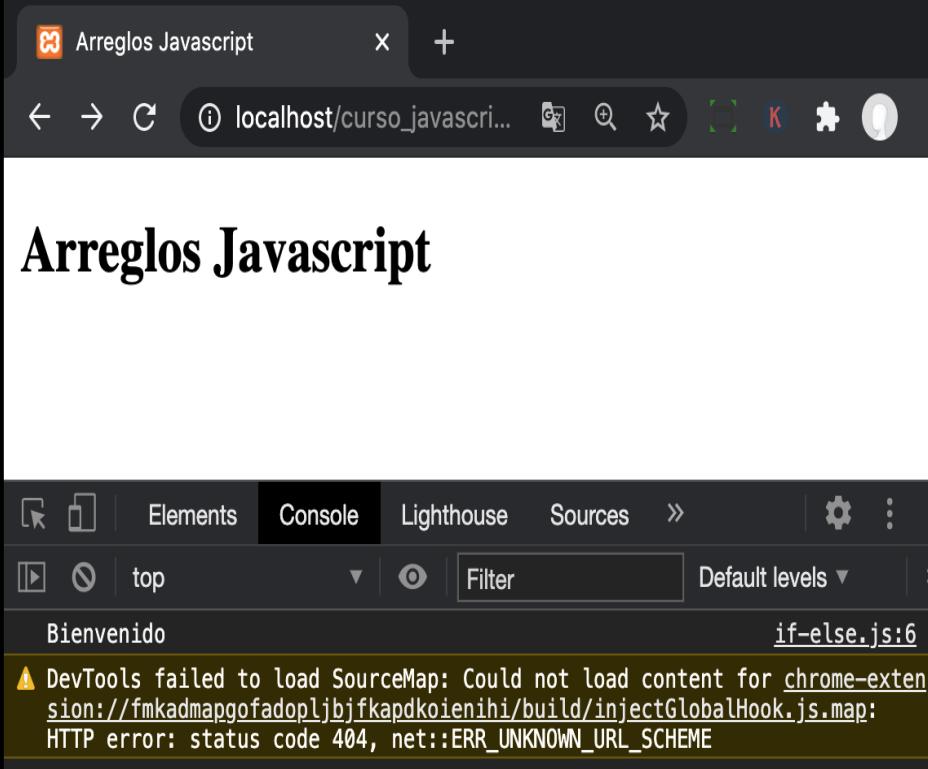
top Filter Default levels

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoiienih/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

a es menor a 10 if-else.js:18

Operadores Lógicos

Operador	Uso	Descripción
AND Lógico (<code>&&</code>)	<code>expr1 && expr2</code>	Devuelve <code>expr1</code> si se puede convertir a <code>false</code> ; de lo contrario, devuelve <code>expr2</code> . Por lo tanto, cuando se usa con valores booleanos, <code>&&</code> devuelve <code>true</code> si ambos operandos son <code>true</code> ; de lo contrario, devuelve <code>false</code> .
OR lógico (<code> </code>)	<code>expr1 expr2</code>	Devuelve <code>expr1</code> si se puede convertir a <code>true</code> ; de lo contrario, devuelve <code>expr2</code> . Por lo tanto, cuando se usa con valores booleanos, <code> </code> devuelve <code>true</code> si alguno de los operandos es <code>true</code> ; si ambos son falsos, devuelve <code>false</code> .
NOT lógico (<code>!</code>)	<code>!expr</code>	Devuelve <code>false</code> si su único operando se puede convertir a <code>true</code> ; de lo contrario, devuelve <code>true</code> .



The image shows a screenshot of a browser developer tools interface, specifically the 'Console' tab. At the top, there is a header bar with the title 'Arreglos Javascript' and a URL 'localhost/curso_javascript...'. Below the header are standard browser navigation buttons (back, forward, search, etc.). The main area displays the text 'Arreglos Javascript' in large, bold, black font.

On the left side of the interface, there is a code editor window titled 'if-else.js'. It contains the following JavaScript code:

```
1 let a=8;
2 let nombre='Rolando';
3
4
5 if(a==8 && nombre==='Rolando'){
6     console.log('Bienvenido');
7 }else{
8     console.log('Hasta Luego');
9 }
10
11
```

The code editor has line numbers on the left and syntax highlighting. A small screenshot of the browser interface is visible in the background of the code editor window.

Operador ternario o if else resumido

A screenshot of a code editor showing a file named `if-else.js`. The code demonstrates the use of the ternary operator (`? :`) as a shorthand for `if...else` statements. It includes a comment explaining the use of the ternary operator.

```
12  /*if(a==10){
13  |     console.log('a es mayor a 10');
14 }else{
15  |     console.log('fin del programa');
16 }*/
17
18 let a=8;
19
20 /*if(a == 10){
21 |     console.log('a es igual a 10');
22 }else if(a > 10){
23 |     console.log('a es mayor a 10');
24 }else if(a < 10){
25 |     console.log('a es menor a 10');
26 }*/
27
28 console.log(a==10 ? 'a igual a 10':'a no igual a 10')
```

A screenshot of a browser developer tools console window titled "Arreglos Javascript". The URL is `localhost/curso_javascript`. The console tab is active, showing the output of the code from the previous screenshot. A warning message about a failed SourceMap load is visible at the bottom.

Arreglos Javascript

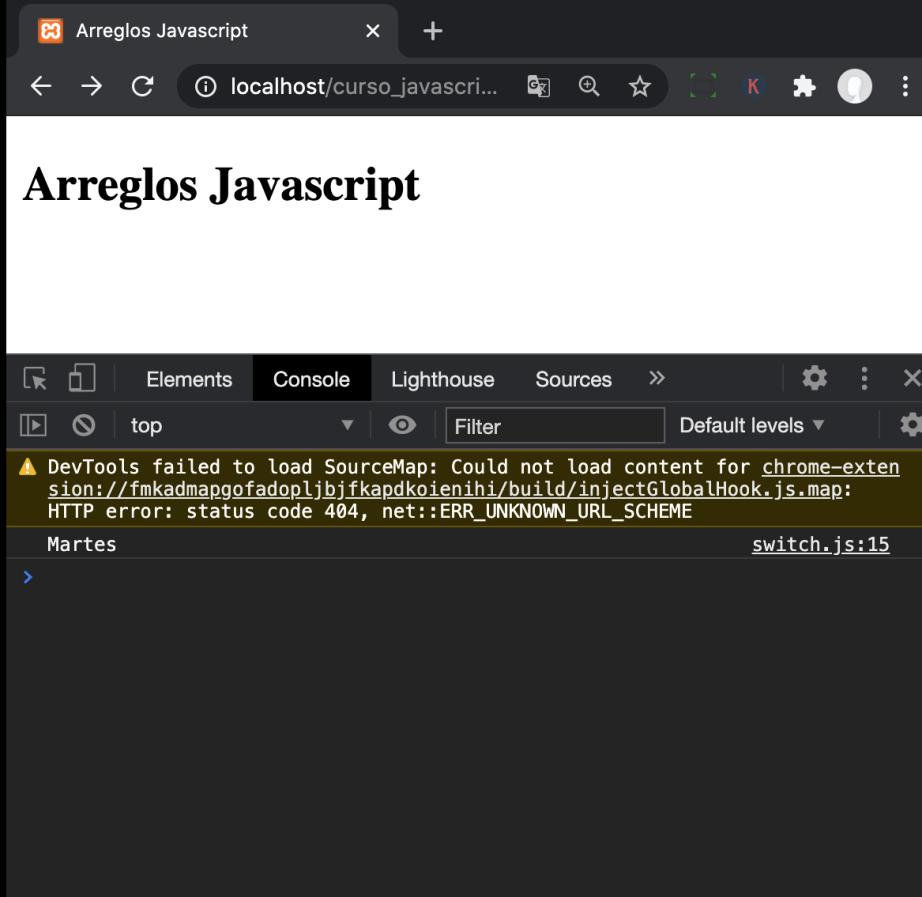
Console

Default levels

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienhi/build/injectGlobalHook.js.map:
HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME

a no igual a 10 if-else.js:28

SWITCH



The screenshot shows a code editor and a browser window. The code editor on the left has tabs for 'index.html' and 'switch.js'. The 'switch.js' tab contains the following code:

```
1 const dia=2;
2
3 switch(dia){
4
5     case 0:
6         console.log('Domingo');
7         break;
8
9     case 1:
10        console.log('Lunes');
11        break;
12
13     case 2:
14         console.log('Martes');
15         break;
16
17     default:
18         console.log('No es domingo, lunes o martes')
19
20 }
21
```

The browser window on the right displays the output of the script: 'Arreglos Javascript' followed by 'Martes'.

DevTools console output:

- ⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
- Martes

REALIZAMOS LOS SIGUIENTES EJERCICIOS

1. CREAR FUNCIONES PROPIAS QUE ESPERE DOS NUMEROS Y DEVUELVA SU SUMA,RESTA, MULTIPLICACION Y DIVISION
2. CREAR UNA FUNCION QUE CALCULE EL AREA DE UN TRIANGULO
3. CREA UNA FUNCION QUE CALCULE EL PERIMETRO DE UN RECTANGULO
4. CREA UNA FUNCION QUE EVALUE 5 DATOS DE EMPLEADOS Y NOS DEVUELVA SU NOMBRE Y QUIEN ES EL QUE TIENE MAYOR SUELDO
5. EN UN HOSPITAL EXISTEN 3 AREAS PEDRIATRIA, URGENCIA, TRAUMATOLOGIA, EL PRESUPUESTO ANUAL SE REPARTE DE LA SIGUIENTE MANERA URGENCIA 37%, PEDRIAGRIA 42%, TRAUMATOLOGIA 21%, CALCULAR LA CANTIDAD DE DINERO QUE RECIBIRA CADA AREA PARA CUALQUIER MONTO PRESUPUESTAL

Clases

Una clase es un molde del que luego se pueden crear múltiples objetos, con similares características. Un objeto se puede caracterizar en una entidad física o una entidad abstracta se compone por tres características principales, estado, comportamiento, identidad. Una clase es una plantilla (molde), que define atributos(variables) y métodos(funciones)

Sintaxis

```
class Persona {  
    constructor(){  
    }  
}
```

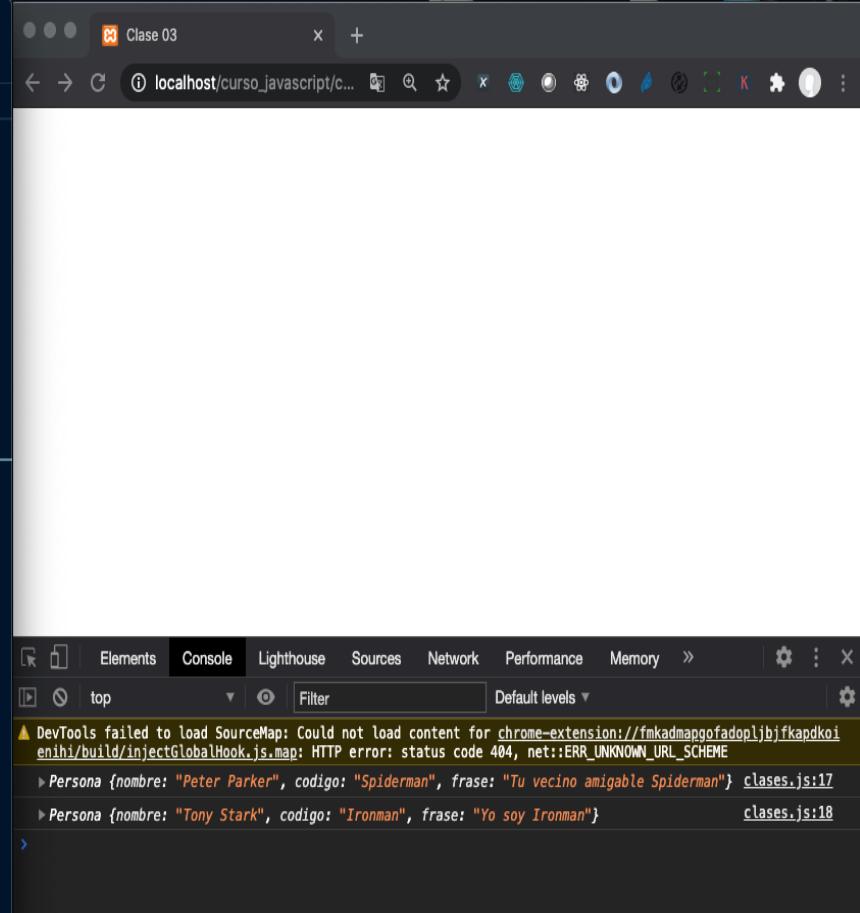
Creamos esta estructura y archivos

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER View:** Shows a folder structure under "CLASE03". The "js" folder contains "clases.js", and the root folder contains "index.html". The "index.html" file is currently selected.
- EDITOR View:** The "index.html" tab is open, displaying the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Clase 03</title>
</head>
<body>
    <script src="js/clases.js"></script>
</body>
</html>
```
- Status Bar:** Shows "Ln 9, Col 30" and "Spaces: 4" as the current code style.

Ejemplo de uso clase



The screenshot shows a code editor and a browser developer tools console. The code editor on the left contains a file named 'clases.js' with the following content:

```
1
2 class Persona {
3
4     nombre = '';
5     codigo = '';
6     frase = '';
7     constructor(nombre = 'Sin nombre',codigo ='Sin codigo',frase='Sin frase'){
8         this.nombre = nombre;
9         this.codigo = codigo;
10        this.frase = frase;
11    }
12 }
13
14 const spiderman = new Persona('Peter Parker','Spiderman','Tu vecino amigable Spiderman');
15 const ironman = new Persona('Tony Stark','Ironman','Yo soy Ironman');
16
17 console.log(spiderman);
18 console.log(ironman);
```

The browser window on the right displays the output of the console.log statements. The developer tools console at the bottom shows the following logs:

- ⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoi/enhi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
- ▶ Persona {nombre: "Peter Parker", codigo: "Spiderman", frase: "Tu vecino amigable Spiderman"} `clases.js:17`
- ▶ Persona {nombre: "Tony Stark", codigo: "Ironman", frase: "Yo soy Ironman"} `clases.js:18`

GET/SET

The screenshot shows a development environment with two main windows. On the left is a code editor with a dark theme, displaying a JavaScript file named `clases.js`. The code defines a `Persona` class with properties `nombre`, `codigo`, `frase`, and `comida`, and methods `setComidaFavorita` and `getComidaFavorita`. A red box highlights the `setComidaFavorita` method. Another red box highlights the `//Get/SET` comment and the assignment `spiderman.setComidaFavorita = 'Tallarín de carne' ;`. On the right is a browser window titled "Clase 03" showing the URL `localhost/curso_javas...`. The browser's developer tools are open, specifically the Console tab, which logs the output of the console statements from the script. The logs show the creation of `spiderman` and `ironman` objects, their properties, and the result of the `getComidaFavorita` method. A red arrow points from the highlighted code in the editor to the corresponding log entry in the browser's console.

```
index.html      Clase 03
js  clases.js  localhost/curso_javas...
js > js clases.js > Persona
1
2  class Persona {
3    nombre = '';
4    codigo = '';
5    frase = '';
6    comida= '';
7    constructor(nombre = 'Sin nombre',codigo ='Sin codigo',frase='Sin frase'){
8      this.nombre = nombre;
9      this.codigo = codigo;
10     this.frase = frase;
11   }
12   set setComidaFavorita(comida){
13     this.comida = comida;
14   }
15
16   get getComidaFavorita(){
17     return this.comida;
18   }
19
20 }
21
22 const spiderman = new Persona('Peter Parker','Spiderman',
23 'Tu vecino amigable Spiderman');
24
25 const ironman = new Persona('Tony Stark','Ironman','Yo soy Ironman');
26
27 console.log(spiderman);
28 console.log(ironman);
29
30 //Get/SET
31 spiderman.setComidaFavorita = 'Tallarín de carne' ;
32 console.log(spiderman.getComidaFavorita);

[clases.js:26] Persona {nombre: "Peter Parker", codigo: "Spiderman", frase: "Tu vecino amigable Spiderman", comida: ""}
[clases.js:27] Persona {nombre: "Tony Stark", codigo: "Ironman", frase: "Yo soy Ironman", comida: ""}
[clases.js:31] Tallarín de carne
[clases.js:31] DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjtkaqdokienihf/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
```

STATIC SE PUEDE ACCEDER A METODOS Y VARIABLES DE TIPO STATIC SIN NECESIDAD DE INSTANCIAR LA CLASE

```
1  class Persona {  
2      nombre = '';  
3      codigo = '';  
4      frase = '';  
5      comida= '';  
6      static _conteo=0;  
7  
8      static get conteo(){  
9          return Persona._conteo + 'instancias';  
10     }  
11     static mensaje() {  
12         console.log('Hola a todos, soy un metodo static');  
13     }  
14  
15     constructor(nombre = 'Sin nombre',codigo ='Sin codigo',frase='Sin frase'){  
16         this.nombre = nombre;  
17         this.codigo = codigo;  
18         this.frase = frase;  
19         Persona._conteo++;  
20     }  
21     set setComidaFavorita(comida){  
22         this.comida = comida;  
23     }  
24  
25     get getComidaFavorita(){  
26         return this.comida;  
27     }  
28  
29 }  
30  
31 const spiderman = new Persona('Peter Parker','Spiderman',  
32 'Tu vecino amigable Spiderman');
```

The screenshot shows a code editor and a browser developer tools console side-by-side.

Code Editor (left):

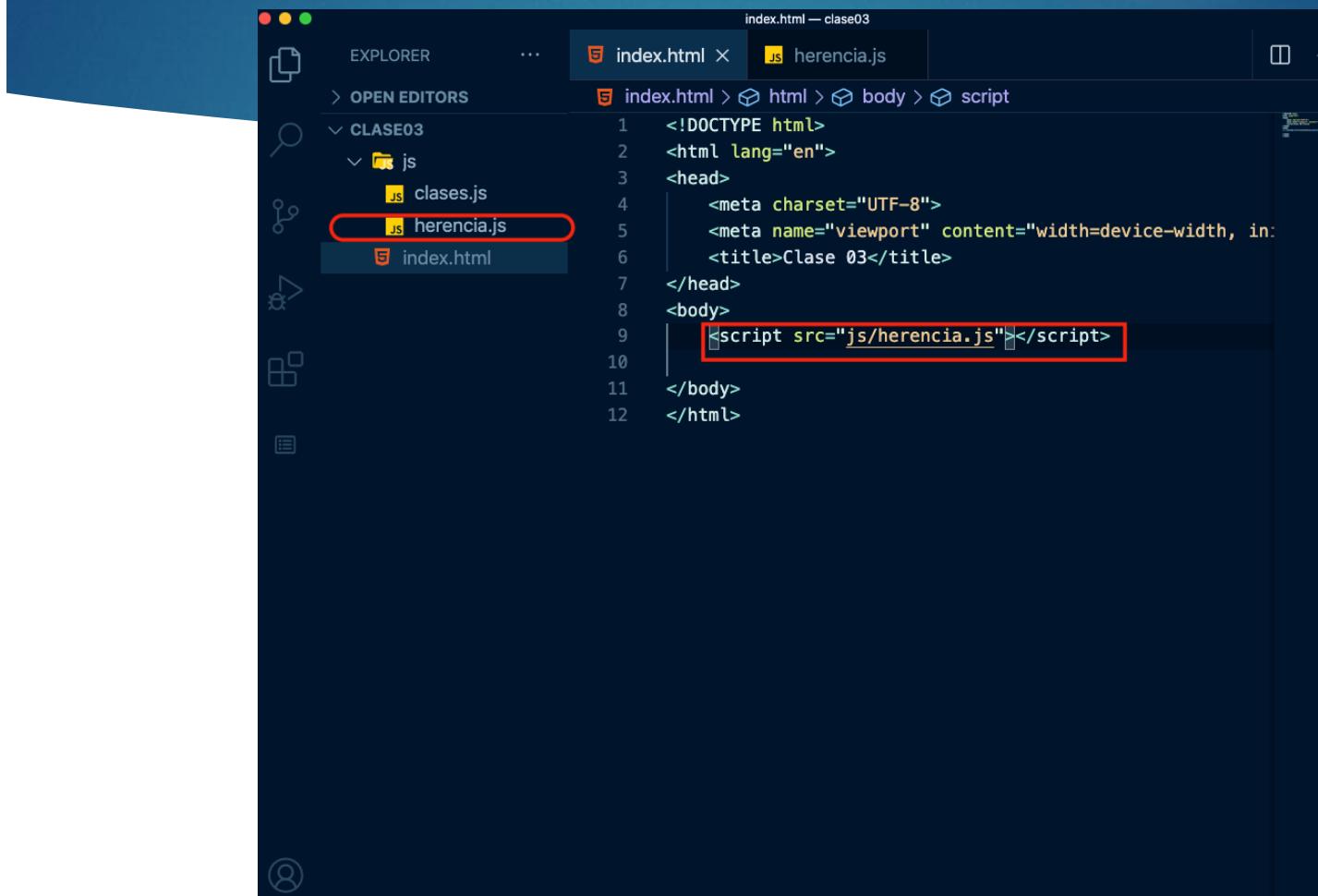
```
index.html    Clase 03
index.html    Clases.js X
js > Clases.js > Persona > (get) conteo
18     this.frase = frase;
19     Persona._conteo++;
20 }
21 set setComidaFavorita(comida){
22     this.comida = comida;
23 }
24
25 get getComidaFavorita(){
26     return this.comida;
27 }
28
29 }
30
31 const spiderman = new Persona('Peter Parker','Spiderman',
32 'Tu vecino amigable Spiderman');
33
34 const ironman = new Persona('Tony Stark','Ironman','Yo soy Ironman');
35
36 console.log(spiderman);
37 console.log(ironman);
38
39 //Get/SET
40 spiderman.setComidaFavorita = 'Tallarín de carne' ;
41 console.log(spiderman.getComidaFavorita);
42
43 // Static
44 console.log(Persona.conteo);
45 Persona.mensaje();
46
47
```

Browser Developer Tools Console (right):

```
Elements  Console  Lighthouse  Sources  Network  >
top  Filter  Default levels
clases.js:36
  ▶ Persona {nombre: "Peter Parker", codigo: "Spiderman", frase: "Tu vecino amigable Spiderman", comida: ""}
clases.js:37
  ▶ Persona {nombre: "Tony Stark", codigo: "Ironman", frase: "Yo soy Ironman", comida: ""}
clases.js:41
  Tallarín de carne
clases.js:44
  2 instancias
clases.js:44
  Hola a todos, soy un metodo static
clases.js:44
  *
```

A red box highlights the code block from line 43 to 46, and a red arrow points from this box to the corresponding output in the developer tools console.

Herencia o Subclases



The screenshot shows a dark-themed code editor interface with the following details:

- EXPLORER** sidebar: Shows a project structure under "CLASE03". The "js" folder contains two files: "clases.js" and "herencia.js". The "herencia.js" file is currently selected and highlighted with a red border.
- OPEN EDITORS** tab bar: Contains tabs for "index.html" (active) and "herencia.js".
- index.html — clase03** editor area:

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, in:
6      <title>Clase 03</title>
7  </head>
8  <body>
9      <script src="js/herencia.js"></script>
10 </body>
11 </html>
```

The line `<script src="js/herencia.js"></script>` is also highlighted with a red border.

The screenshot shows a code editor and a browser window side-by-side.

Code Editor (herencia.js):

```
index.html    herencia.js ×
js > herencia.js > ...
21     set setComidaFavorita(comida){
22         this.comida = comida;
23     }
24
25     get getComidaFavorita(){
26         return this.comida;
27     }
28
29 }
30
31 class Heroe extends Persona{
32     clan = 'Sin clan';
33
34     constructor(nombre,codigo,frase){
35         //palabra reservada que hace referencia al
36         //constructor padre
37         super(nombre,codigo,frase);
38         this.clan='Advangers';
39     }
40
41 }
42
43 const hulk = new Heroe('Banner','Hulk','Hulk Aplasta','advengers');
44 console.log(hulk);
45 hulk.setComidaFavorita = 'asado a la estaca';
46 console.log(hulk.getComidaFavorita());
47
```

The code defines a `Heroe` class that extends `Persona`. It includes a constructor that calls `super`, setting the `clan` to 'Advangers'. It also includes a `setComidaFavorita` method and a `getComidaFavorita` method. Finally, it creates a `hulk` object and logs its properties to the console.

Browser Window (Clase 03):

The browser window shows the output of the code execution:

```
Clase 03
localhost/curso_javas...
Default levels ▾
DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadopljbjfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
herencia.js:45
> Heroe {nombre: "Banner", codigo: "Hulk", frase: "Hulk Aplasta", comida: "", clan: "Advangers"}
asado a la estaca
herencia.js:47
> *
```

The browser displays the object created (`Heroe`), its properties (`nombre`, `codigo`, `frase`, `comida`, `clan`), and the value assigned to `comida`.

Promesas

The screenshot shows a dark-themed code editor interface, likely Visual Studio Code, with the following details:

- Explorer View:** On the left, under the "CLASE03" folder (which contains a "js" folder), the "index.html" file is currently selected.
- Editor Area:** The main area displays the content of "index.html".

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, in:
        <title>Clase 03</title>
</head>
<body>
    <script src="js/promesas.js"></script>
</body>
</html>
```
- File Tabs:** At the top, there are tabs for "promesas.js" and "index.html", with "index.html" being the active tab.
- Code Editor Status:** A status bar at the bottom indicates "Line 1, Col 1" and "12 lines".

promesas.js — clase03

promesas.js X index.html

js > js promesas.js > [o] promesa > <function>

```
1 //nos permite esperar el tiempo que sea por una tarea
2 const promesa = new Promise((resolve,reject) =>[
3
4     setTimeout(()=>{
5         const exito = true;
6         if(exito){
7             resolve();
8         }else{
9             reject();
10        }
11    }
12
13    ,4000);
14
15
16
17 ]);
18
19 promesa.then(()=>{
20     alert('Exito');
21 });
22
23 promesa.catch(()=>{
24     alert('No exitosa');
25 });
```

localhost dice

Exito

Aceptar

Elements Console Lighthouse Sources Network >

top Filter Default levels ▾

⚠ DevTools failed to load SourceMap: Could not load content for chrome-extension:gofadoplbjbfkapdkoienihi/build/injectGlobalHook.js.map: HTTP error: status code net::ERR_UNKNOWN_URL_SCHEME

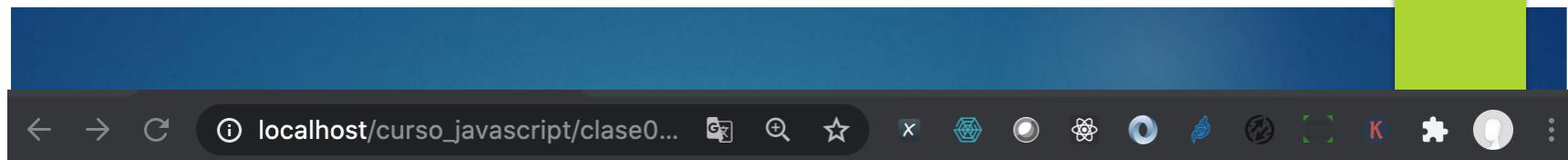
DOM

- ▶ Document Object Model o lo que es lo mismo, la estructura del documento HTML
- ▶ En Javascript, cuando nos referimos al **DOM** nos referimos a esta estructura, que podemos modificar de forma dinámica desde Javascript, añadiendo nuevas etiquetas, modificando o eliminando otras, cambiando sus atributos HTML, añadiendo clases, cambiando el contenido de texto, etc

The screenshot shows a code editor interface with a dark theme. The left sidebar contains icons for Explorer, Search, Open Editors, File Explorer, Find, Diff, and Help. The main area has tabs for 'index.html' and 'dom'. The 'dom' tab is active, showing a tree view of the DOM structure: dom > index.html > html > body > script. Below this, the code editor displays the following HTML and JavaScript:

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <script src="js/app.js"></script>
8  </head>
9  <body>
10     <div>
11         <h1 id="titulo">Hola soy el titulo</h1>
12         <p id = "parrafo">Lorem ipsum dolor sit amet, consectetur adipisicing</p>
13         <button id="boton" onclick="alert('click')">Cambiar</button>
14     </div>
15     <script>
16         let boton = document.getElementById("boton");
17
18         function cambiar(){
19             let h1 = document.getElementById("titulo");
20             console.log(h1.innerHTML);
21             h1.style.color="red";
22         }
23
24         boton.onclick = cambiar;
25
26     </script>
27
28 </body>
29 </html>
```

The code includes an HTML structure with a title, a paragraph, and a button. The button's onclick event calls a JavaScript function named 'cambiar'. This function logs the current innerHTML of the 'h1' element and then changes its style color to red. The script is included via a `<script>` tag in the head.



Hola soy el titulo

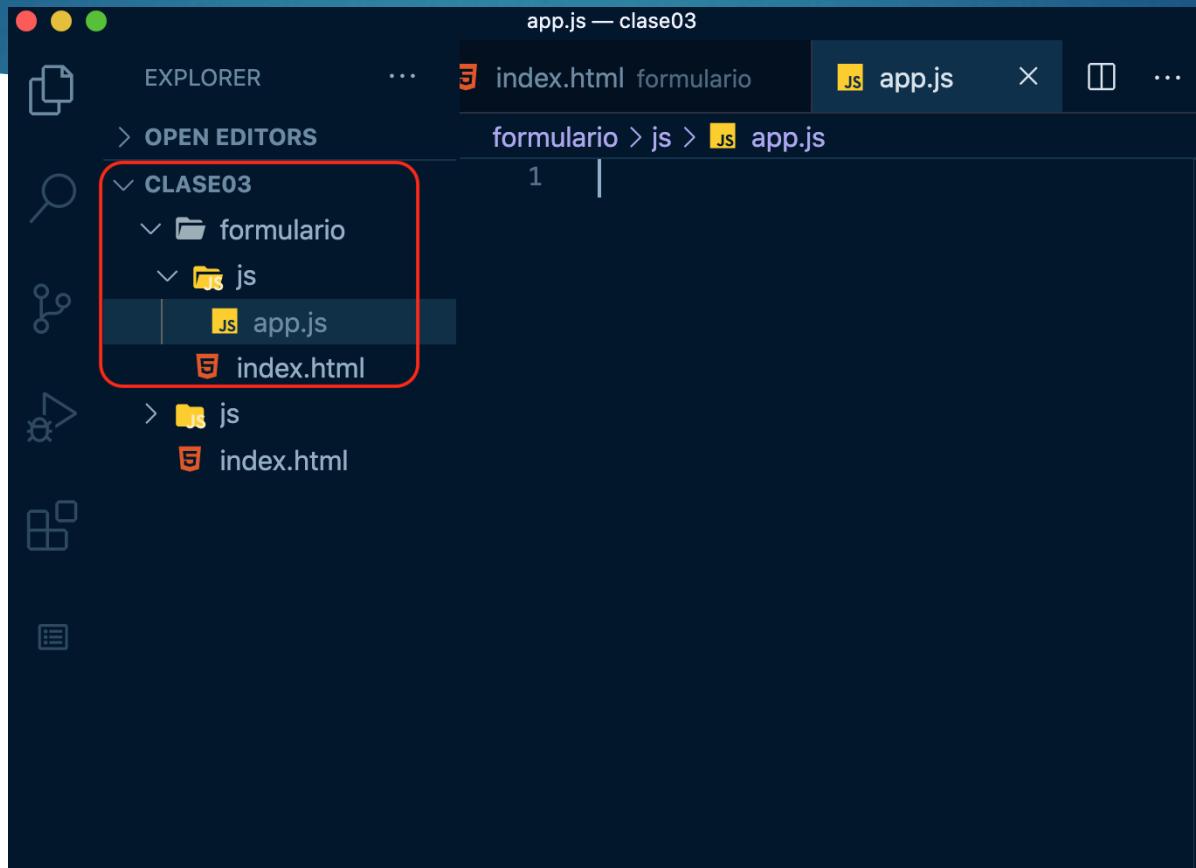
Lorem ipsum dolor sit amet, consectetur adipisicing

[Cambiar](#)

The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The console output is as follows:

```
DevTools failed to load SourceMap: Could not load content for chrome-extension://fmkadmapgofadopljbjfkapdkoiienih_i/build/injectGlobalHook.js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
Hola soy el titulo
(index):20
```

Formularios html en Javascript



index.html

JS app.js

X



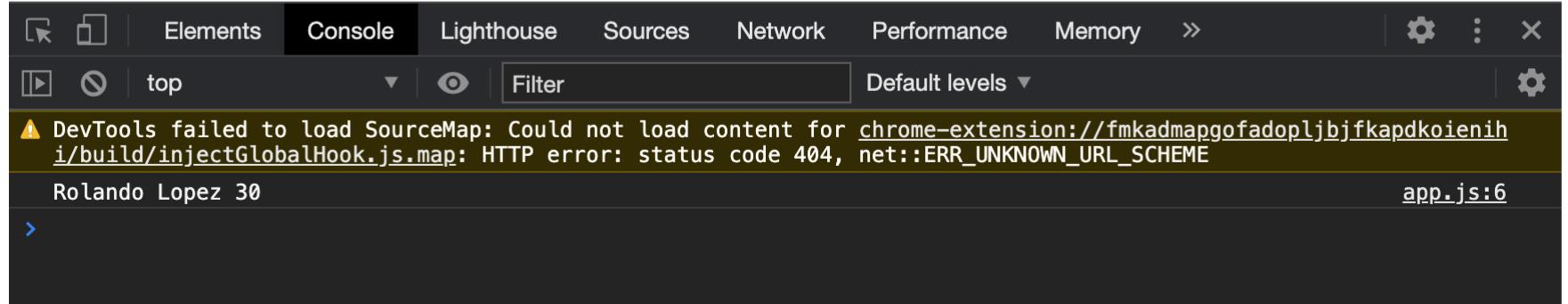
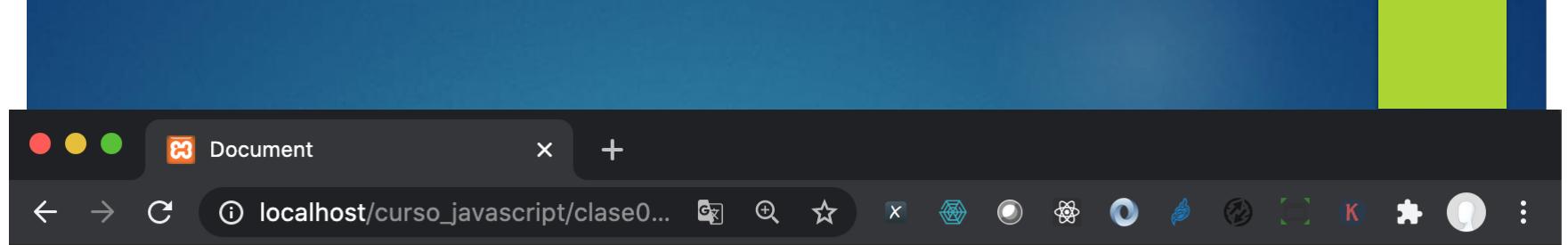
formulario > js > JS app.js > ...

```
1
2 let getData = function (){
3     let name = document.getElementById("name").value;
4     let age = document.getElementById("age").value;
5
6     console.log(name+' '+age);
7 }
8
9 
```

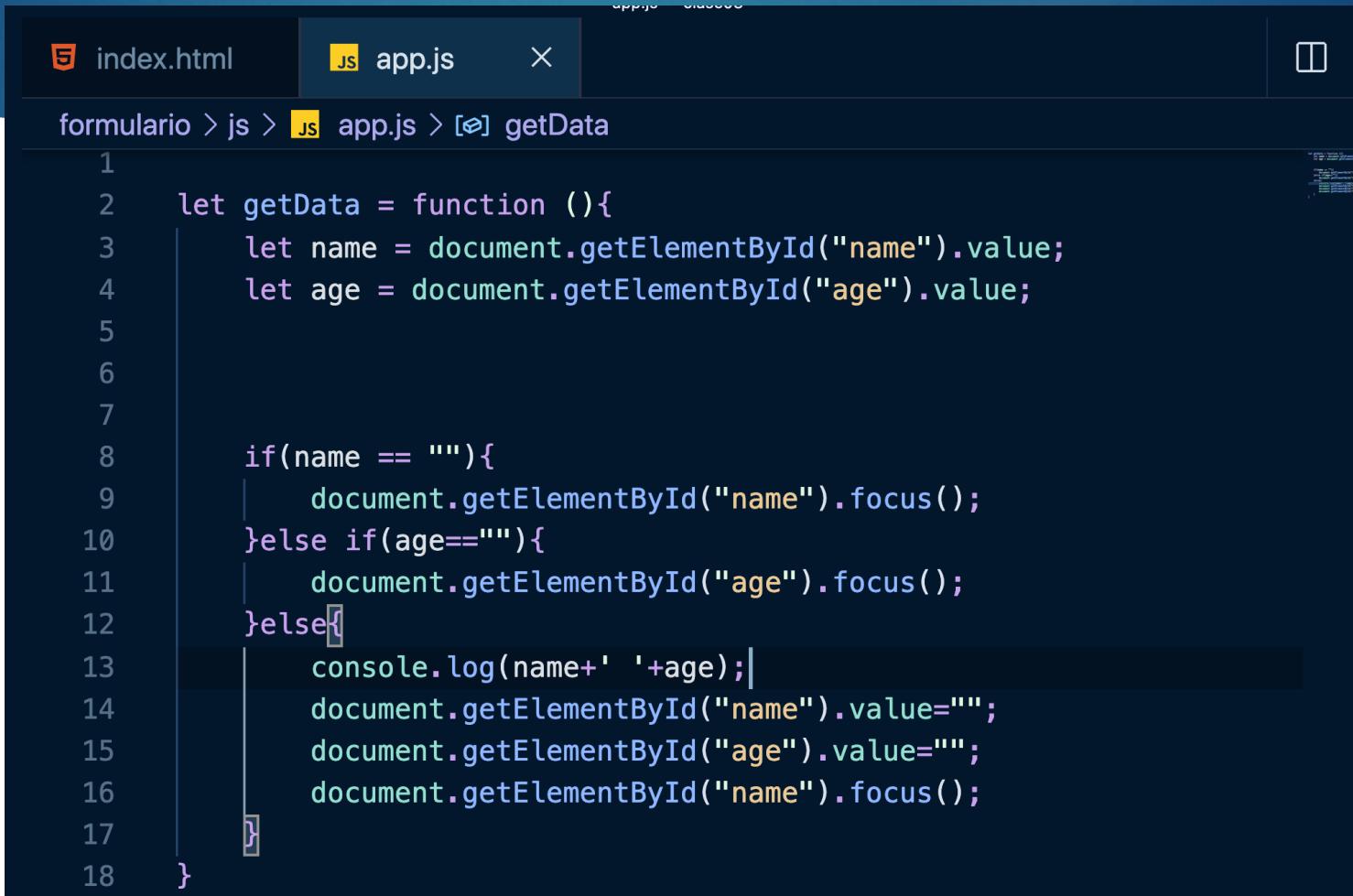
index.html X app.js

formulario > index.html > html > head > script

```
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7      <script src="js/app.js"></script>
8    </head>
9    <body>
10      <form action="">
11        <input type="text" name="name" id="name" placeholder="nombre" ><br>
12        <input type="text" name="age" id="age" placeholder="Año"><br>
13        <button type="button" onclick="getData()">Enviar</button>
14      </form>
15
16    </body>
17  </html>
```



Mejoramos nuestro código



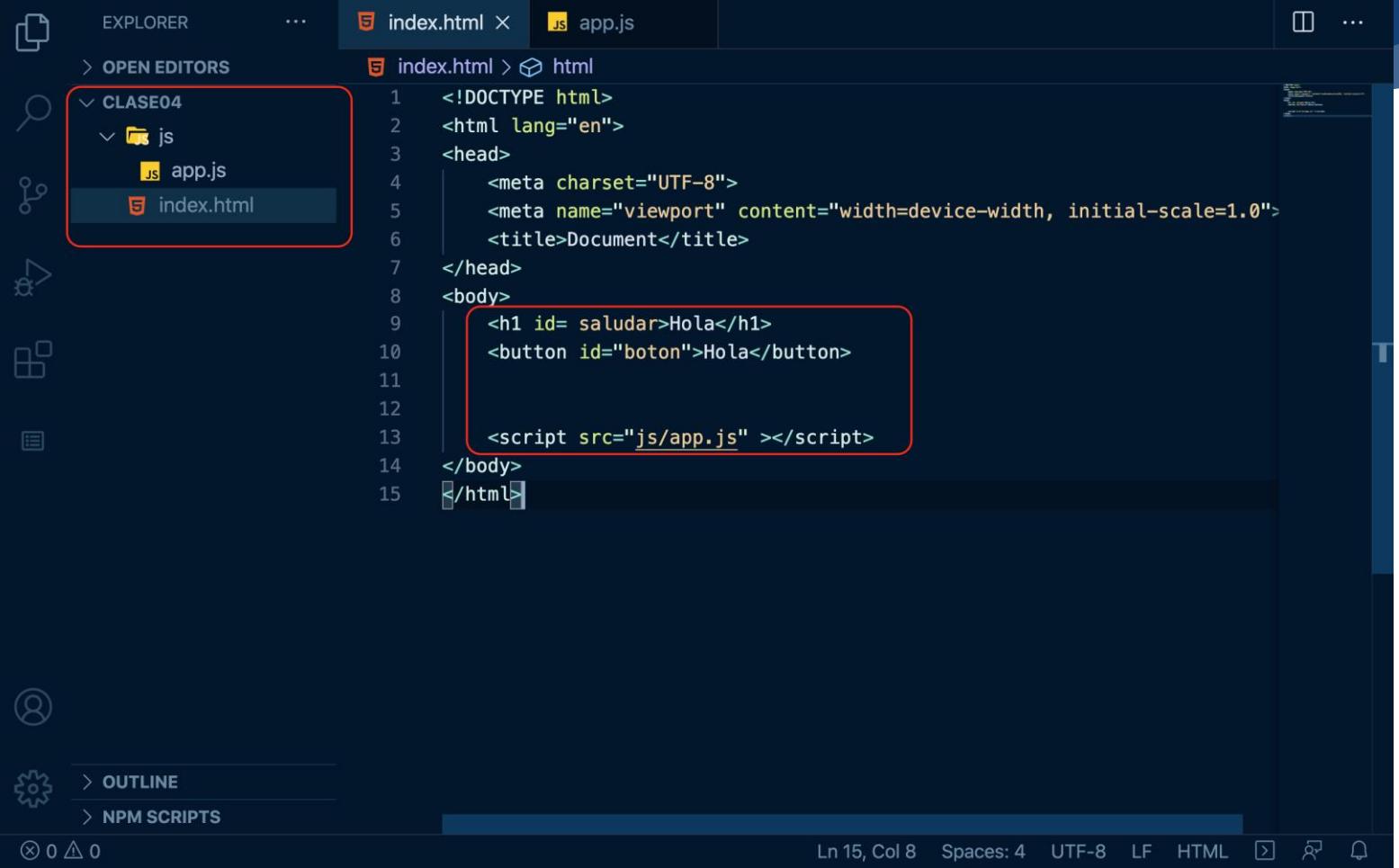
The screenshot shows a code editor interface with two tabs: "index.html" and "app.js". The "app.js" tab is active, displaying the following JavaScript code:

```
1
2  let getData = function (){
3      let name = document.getElementById("name").value;
4      let age = document.getElementById("age").value;
5
6
7
8      if(name == ""){
9          document.getElementById("name").focus();
10     }else if(age==""){
11         document.getElementById("age").focus();
12     }else{
13         console.log(name+' '+age);
14         document.getElementById("name").value="";
15         document.getElementById("age").value="";
16         document.getElementById("name").focus();
17     }
18 }
```

The code defines a function named `getData` that retrieves values from two input fields, `name` and `age`. If either field is empty, it focuses on the respective input element. Otherwise, it logs the concatenated values to the console and clears both fields before focusing on the `name` input again.

Más Elementos DOM

Creamos esta estructura y archivos



The screenshot shows the VS Code interface with the following details:

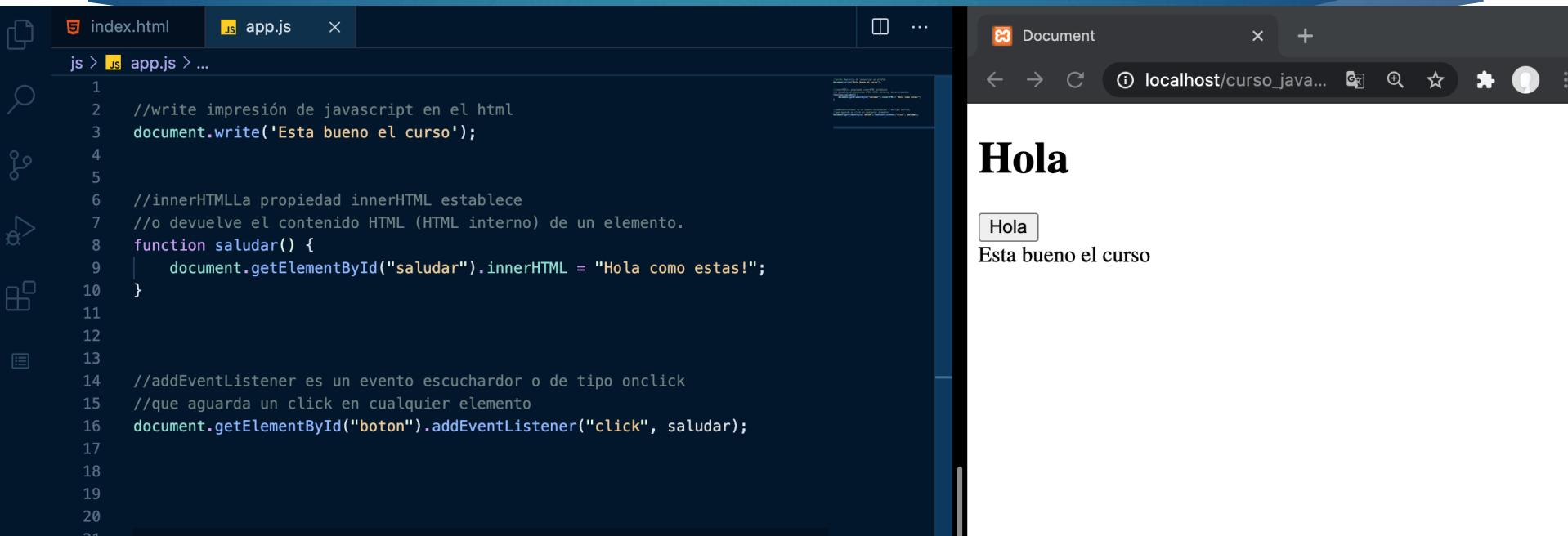
- EXPLORER** view: Shows a folder structure under "OPEN EDITORS". A red box highlights the "CLASE04" folder, which contains a "js" folder and files "app.js" and "index.html".
- index.html** editor tab: The code editor displays the following HTML content:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1 id="saludar">Hola</h1>
    <button id="boton">Hola</button>

    <script src="js/app.js" ></script>
</body>
</html>
```

A red box highlights the script tag.
- Bottom status bar:** Shows "Ln 15, Col 8" and other file status indicators.

Ejemplo de Write, innerHTML y addEventListener



The image shows a code editor on the left and a browser window on the right. The code editor displays the file `app.js` with the following content:

```
index.html | JS app.js | ...
js > JS app.js > ...
1
2 //write impresión de javascript en el html
3 document.write('Esta bueno el curso');
4
5
6 //innerHTML La propiedad innerHTML establece
7 //o devuelve el contenido HTML (HTML interno) de un elemento.
8 function saludar() {
9     document.getElementById("saludar").innerHTML = "Hola como estas!";
10 }
11
12
13
14 //addEventListener es un evento escuchardor o de tipo onclick
15 //que aguarda un click en cualquier elemento
16 document.getElementById("boton").addEventListener("click", saludar);
17
18
19
20
21
```

The browser window shows the URL `localhost/curso_java...`. The page content includes the word **Hola** in a large font, followed by a button labeled **Hola**, and the text **Esta bueno el curso**.

The screenshot shows a code editor interface with two tabs: "index.html" and "app.js". The "index.html" tab is active, displaying the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
        <title>Document</title>
</head>
<body>
    <h1 id= saludar>Hola</h1>
    <button id="boton">Hola</button>
    <div id="dias"></div>
    <script src="js/app.js" ></script>
</body>
</html>
```

A red oval highlights the line containing the `<div id="dias"></div>` element. The code editor has a dark theme with light-colored syntax highlighting. On the left side, there is a vertical bar of icons representing different file types and project components.

The image shows a development environment with a code editor and a browser preview.

Code Editor (app.js):

```
1 //innerHTML La propiedad innerHTML establece
2 //o devuelve el contenido HTML (HTML interno) de un elemento.
3
4 function saludar() {
5     document.getElementById("saludar").innerHTML = "Hola como estas!";
6 }
7
8
9 //addEventListener es un evento escuchador o de tipo onclick
10 //que aguarda un click en cualquier elemento
11 document.getElementById("boton").addEventListener("click", saludar);
12
13
14
15 const dias=['lunes','martes','miercoles','jueves','viernes',
16 'sabado','domingo'];
17
18 for (let i = 0; i < dias.length; i++) {
19     const element = dias[i];
20     document.getElementById("dias").innerHTML += "<li>" + dias[i] + "</li>"
21 }
22
23
24 }
```

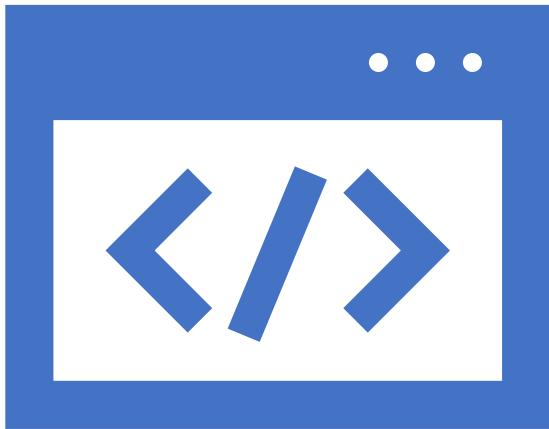
Browser Preview:

The browser window displays the word "Hola" followed by a list of days of the week:

Hola

Hola

- lunes
- martes
- miercoles
- jueves
- viernes
- sabado
- domingo



Posiciones del
`<script></script>` en
HTML

Aplicando lo aprendido para el ejercicio

The screenshot shows a code editor interface with the following details:

- EXPLORER** view: Shows a tree structure of files and folders under "CLASE04". A red box highlights the "ejercicio1" folder, which contains "index.html" and "js" (containing "ejercicio1.js"). Other visible items include "fetch" and another "js" folder.
- OPEN EDITORS** tab: Shows two open files: "index.html" and "ejercicio1.js".
- File Structure:** The "index.html" file is structured as follows:
 - <!DOCTYPE html>
 - <html lang="en">
 - <head>
 - <meta charset="UTF-8">
 - <meta name="viewport" content="width=device-width, initial-scale=1.0">
 - <title>Document</title>
 - </head>
 - <body>
 - <h1>Realize las Operaciones</h1>
 - <input type="number" id="num1">

 - <input type="number" id="num2">

 -

 - <button id="boton1">Sumar</button>
 - <button id="boton2">Resta</button>
 - <button id="boton3">Multiplicación</button>
 - <button id="boton4">División</button>
 -

 - <h2 id="respuesta">Respuesta</h2>
 - <script src="js/ejercicio1.js"></script>
 - </body>
 - </html>
- Code Editor:** The "ejercicio1.js" file content is shown below the file structure:

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7
8  </head>
9  <body>
10     <h1>Realize las Operaciones</h1>
11     <input type="number" id="num1"> <br>
12     <input type="number" id="num2"> <br>
13
14     <button id="boton1">Sumar</button>
15     <button id="boton2">Resta</button>
16     <button id="boton3">Multiplicación</button>
17     <button id="boton4">División</button>
18
19     <br>
20     <h2 id="respuesta">Respuesta</h2>
21     <script src="js/ejercicio1.js"></script>
22
23 </body>
24 </html>
```

The screenshot shows a code editor interface with the following details:

- Explorer View:** Shows a tree structure of files and folders. The folder "CLASE04" contains "ejercicio1" which has a "js" folder containing "ejercicio1.js". Other files like "index.html" and "fetch" are also listed.
- Editor Area:** The current file being edited is "ejercicio1.js". The code implements four event listeners for buttons "boton1" through "boton4". Each button's click event triggers a function that reads two input numbers from HTML elements "num1" and "num2", performs a calculation (sum, difference, multiplication, or division), and updates a response element with the result.
- Bottom Navigation:** Includes links for "OUTLINE" and "NPM SCRIPTS".

```
1  document.getElementById("boton1").addEventListener("click", function(){
2      let num1 = document.getElementById("num1").value;
3      let num2 = document.getElementById("num2").value;
4      let suma= parseInt(num1)+parseInt(num2);
5      document.getElementById("respuesta").innerHTML = 'La suma es: '+ suma;
6  });
7
8
9
10 document.getElementById("boton2").addEventListener("click", function(){
11     let num1 = document.getElementById("num1").value;
12     let num2 = document.getElementById("num2").value;
13     let resta= parseInt(num1)-parseInt(num2);
14     document.getElementById("respuesta").innerHTML = 'La resta es: '+ resta;
15 });
16
17 document.getElementById("boton3").addEventListener("click", function(){
18     let num1 = document.getElementById("num1").value;
19     let num2 = document.getElementById("num2").value;
20     let multi= parseInt(num1)*parseInt(num2);
21     document.getElementById("respuesta").innerHTML = 'La multiplicación es: '+ multi;
22 });
23
24 document.getElementById("boton4").addEventListener("click", function(){
25     let num1 = document.getElementById("num1").value;
26     let num2 = document.getElementById("num2").value;
27     let divi= parseInt(num1)/parseInt(num2);
28     document.getElementById("respuesta").innerHTML = 'La división es: '+ divi;
29 });
30
31
32
```

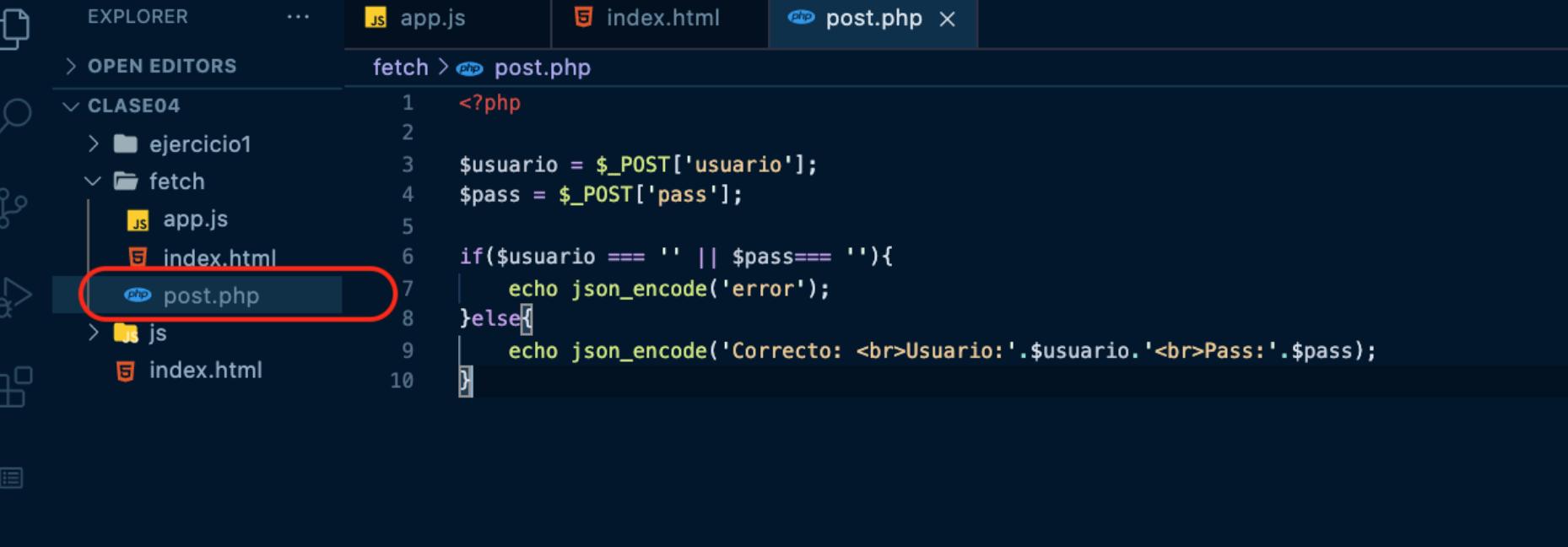
Uso de Fetch en Javascript

The screenshot shows a code editor interface with the following details:

- EXPLORER**: Shows a file tree with the following structure:
 - CLASE04
 - ejercicio1
 - fetch
 - app.js
 - index.html
 - post.php
 - index.html
- OPEN EDITORS**: Shows two open files:
 - app.js
 - index.html
- index.html** (active tab):

```
> fetch > index.html > html > body > div.container.my-5 > div#respuesta.mt-3

1  <!doctype html>
2  <html lang="en">
3
4  <head>
5      <!-- Required meta tags -->
6      <meta charset="utf-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8
9      <!-- Bootstrap CSS -->
10     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" integrity="sha384-WskhaSGFgHYwDcbwN70/dfYBj47jz9qbsMId/iRN3ewGhXQFZCSftd1LZCfmhktb"
11         crossorigin="anonymous">
12
13     <title>Hello, world!</title>
14 </head>
15
16 <body>
17     <div class="container my-5">
18         <h2>Formulario con FetchAPI</h2>
19         <form id="formulario">
20             <input type="text" name="usuario" placeholder="Ingresa usuario" class="form-control my-3">
21             <input type="password" name="pass" placeholder="Ingresa contraseña" class="form-control my-3">
22             <button class="btn btn-primary" type="submit">Enviar</button>
23         </form>
24         <div class="mt-3" id="respuesta">
25             </div>
26         </div>
27     </div>
28
29     <!-- Optional JavaScript -->
30     <!-- jQuery first, then Popper.js, then Bootstrap JS -->
31     <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo"
32         crossorigin="anonymous"></script>
33     <script src="https://cdn.jsdelivr.net/npm/popper.js@1.14.3/dist/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLAdn689aCwoqbBJiSnjAK/l8WvCwPIPm49"
34         crossorigin="anonymous"></script>
35     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyMz4qpPeasjB/pTJ0euyQp0Mk8ck+5T"
36         crossorigin="anonymous"></script>
37     <script src="app.js"></script>
38
39 </body>
40 </html>
```



The screenshot shows the Microsoft Visual Studio Code (VS Code) interface. The top navigation bar includes tabs for "EXPLORER", "...", "app.js", "index.html", and "post.php". The "post.php" tab is currently active, indicated by a blue background and white text. Below the tabs, the "OPEN EDITORS" section shows a tree view of files under the "CLASE04" workspace. The "fetch" folder contains "app.js", "index.html", and "post.php", which is highlighted with a red oval. Other files in the workspace include "ejercicio1", "js", and another "index.html". The main editor area displays the contents of the "post.php" file:

```
1 <?php
2
3 $usuario = $_POST['usuario'];
4 $pass = $_POST['pass'];
5
6 if($usuario === '' || $pass === ''){
7     echo json_encode('error');
8 }else{
9     echo json_encode('Correcto: <br>Usuario:'.$usuario.'<br>Pass:'.$pass);
10 }
```

EXPLORER ...

OPEN EDITORS

CLASE04

> ejercicio1

fetch

> app.js

index.html

post.php

> js

index.html

fetch > app.js > ...

```
1 var formulario = document.getElementById('formulario');
2 var respuesta = document.getElementById('respuesta');
3
4 formulario.addEventListener('submit', function(e){
5     e.preventDefault();
6     console.log('me diste un click')
7
8     var datos = new FormData(formulario);
9
10    console.log(datos);
11    console.log(datos.get('usuario'));
12    console.log(datos.get('pass'));
13
14    fetch('post.php',{
15        method: 'POST',
16        body: datos
17    })
18        .then( res => res.json())
19        .then( data => {
20            console.log(data)
21            if(data === 'error'){
22                respuesta.innerHTML = `
23                    <div class="alert alert-danger" role="alert">
24                        Llena todos los campos
25                    </div>
26
27            }else{
28                respuesta.innerHTML = `
29                    <div class="alert alert-primary" role="alert">
30                        ${data}
31                    </div>
32
33            }
34        }
35    )
36});
```

The image shows a split-screen development environment. On the left, a code editor displays `app.js` with the following content:

```
app.js — clase04
fetch('post.php', {
    method: 'POST',
    body: datos
})
.then( res => res.json())
.then( data => {
    console.log(data)
    if(data === 'error'){
        respuesta.innerHTML = `
        <div class="alert alert-danger" role="alert">
        | Llena todos los campos
        </div>
        `

    }else{
        respuesta.innerHTML = `
        <div class="alert alert-primary" role="alert">
        ${data}
        </div>
        `
    }
});
```

On the right, a web browser window titled "Hello, world!" shows a form with two fields: "qwqw" and "Ingrresa contraseña". A blue button labeled "Enviar" is below the fields. A red callout box with the text "Llena todos los campos" is positioned over the second field.

► **JavaScript Asíncrono + XML (AJAX)** no es una tecnología por sí misma, es un término que describe un nuevo modo de utilizar conjuntamente varias tecnologías existentes. Esto incluye: HTML o XHTML, CSS, JavaScript, DOM, XML, XSLT, y lo más importante, el objeto XMLHttpRequest. Cuando estas tecnologías se combinan en un modelo AJAX, es posible lograr aplicaciones web capaces de actualizarse continuamente sin tener que volver a cargar la página completa. Esto crea aplicaciones más rápidas y con mejor respuesta a las acciones del usuario.

Consumir api rest

- ▶ <https://randomuser.me/>

js app.js index.html X

ajax > index.html > html > body

```
1  <!doctype html>
2  <html lang="en">
3
4  <head>
5      <!-- Required meta tags -->
6      <meta charset="utf-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8
9      <!-- Bootstrap CSS -->
10     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"
11         crossorigin="anonymous">
12
13     <title>Hello, world!</title>
14 </head>
15
16 <body>
17     <div class="container my-5 text-center">
18         <button class="btn btn-danger w-100" onclick="traer()">Obtener</button>
19         <div class="mt-5" id="contenido">
20
21             </div>
22         </div>
23
24         <script src="js/app.js">
25
26     </script>
27
28
29 </body>
30
31 </html>
```

JS app.js X index.html

ajax > js > JS app.js > ...

```
1  var contenido = document.querySelector('#contenido')
2  function traer() {
3      fetch('https://randomuser.me/api/')
4          .then(res => res.json())
5          .then(data => {
6              console.log(data.results['0']);
7              console.log(data.results['0'].email);
8              /*contenido.innerHTML =
9               
10              <p>Nombre: ${data.results['0'].name.last}</p>
11              `*/
12          })
13      }
```

**Gracias por
vuestra
participación**



¡Seguimos en contacto!

www.iconotc.com

