



# UT01: INTRODUCCIÓN A JAVASCRIPT

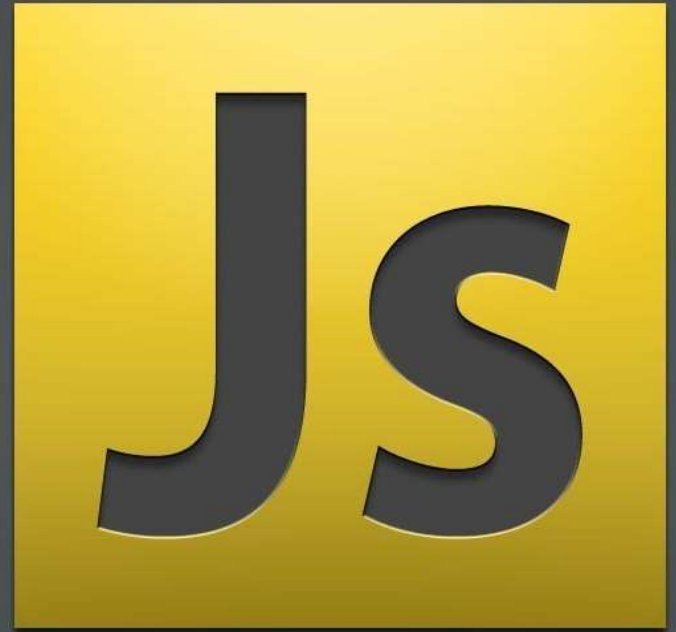
# ÍNDICE

---

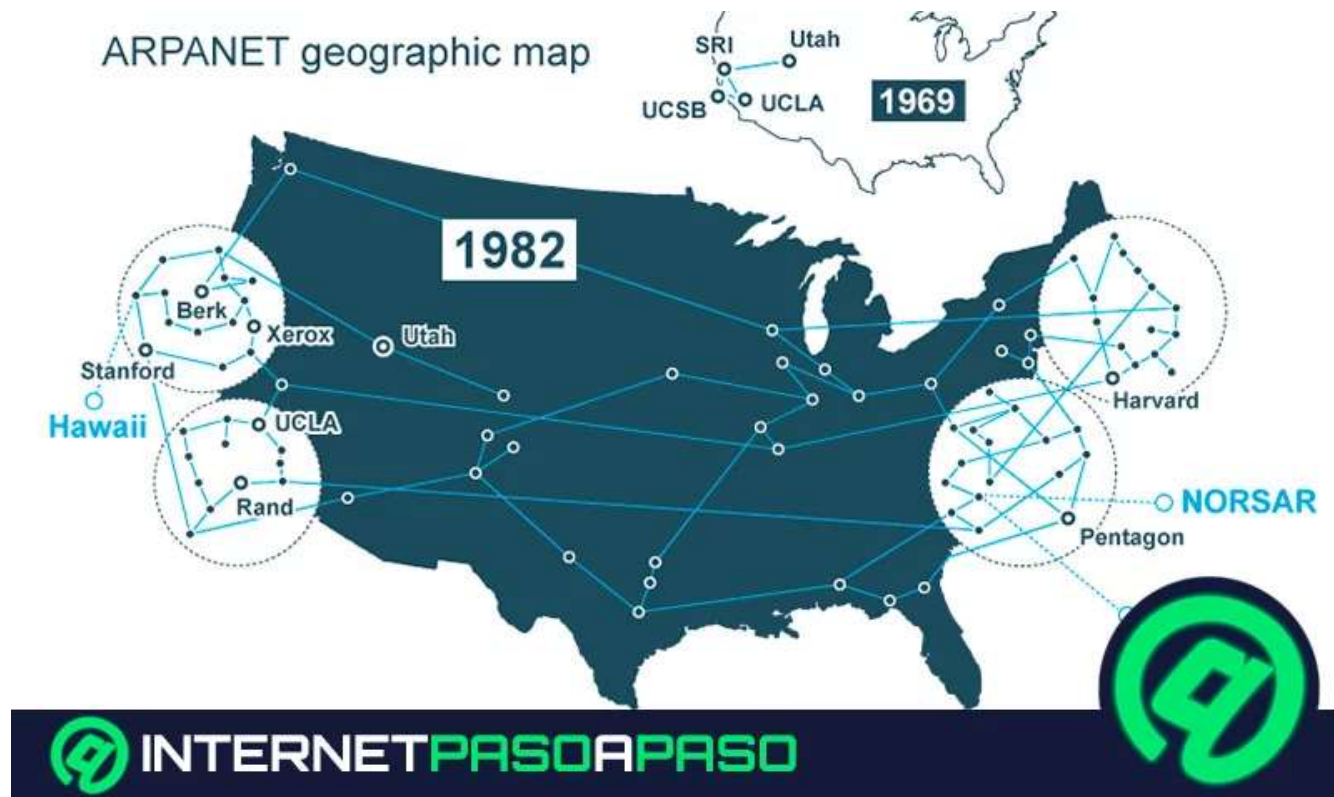
- 1.- Historia de la Web y JavaScript
- 2.- Características de JavaScript
- 3.- Preparación del entorno
- 4.- Ejecución de JavaScript
- 5.- Depuración del código
- 6.- Transpiladores y polyfills

# 1

## HISTORIA DE LA WEB Y JAVASCRIPT



El origen de Internet se remonta al año 1969, cuando el DoD de EEUU desarrolló una red que interconectaba varias universidades. Esta red se denominó **ARPANET**.



En principio **no hab́a ṕginas Web** sino que se utilizaban aplicaciones (en modo texto) para acceder al correo electŕnico, a servidores de ficheros (FTP) o BBS (Bulletin Board System).

```
Monochrome (1.101w 07-May-08) (Last on Wed May 14 13:36)

New streamlined layout! Easier to use! New files! Extra exclamation marks!

Dish some dirt at <MT0> today!

Menu [ESC] = Utilities (inc. Talker & EXIT)

You don't use ssh. Booo! Menu [I] = Help and Information on Monochrome

Welcome to the new version of Monochrome! (version 1.101w)
Menu [N] = News and Media
Menu [T] = Science, Technology and Medicine
Menu [E] = Entertainment
Menu [C] = Society and Culture
Menu [R] = Recreation

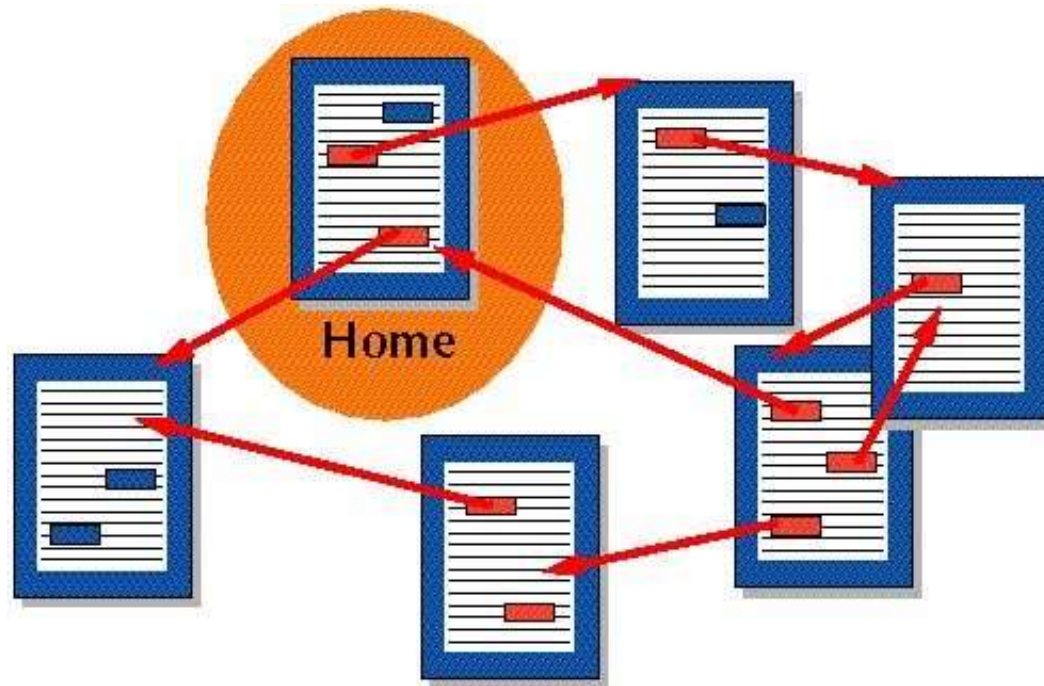
Menu [M] = Monochrome Users

Hello 'SexDrugs&DrumMachinesForAgRaveGeneration'. (evilandi:4)
<< 22 other users at Sun Jan 11 19:30 BST >>
```

La persona que lo cambió todo fue **Tim Berners-Lee**, un científico inglés que trabajaba en el CERN (Organización Europea para la Investigación Nuclear).



Tim Berners-Lee desarrolló un sistema de comunicación, al que denominó **World Wide Web** que estaba basado en **hipertexto: palabras dentro de un documento que enlazaban a otros documentos.**





La primera página Web fue creada el 6 de agosto de 1991 y aún sigue disponible en los servidores del CERN (<http://info.cern.ch/hypertext/WWW/TheProject.html>)

## World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#), [Policy](#), November's [W3 news](#), [Frequently Asked Questions](#).

### [What's out there?](#)

Pointers to the world's online information, [subjects](#), [W3 servers](#), etc.

### [Help](#)

on the browser you are using

### [Software Products](#)

A list of W3 project components and their current state. (e.g. [Line Mode](#), [X11 Viola](#), [NeXTStep](#), [Servers](#), [Tools](#), [Mail robot](#), [Library](#).)

### [Technical](#)

Details of protocols, formats, program internals etc

### [Bibliography](#)

Paper documentation on W3 and references.

### [People](#)

A list of some people involved in the project.

### [History](#)

A summary of the history of the project.

### [How can I help?](#)

If you would like to support the web..

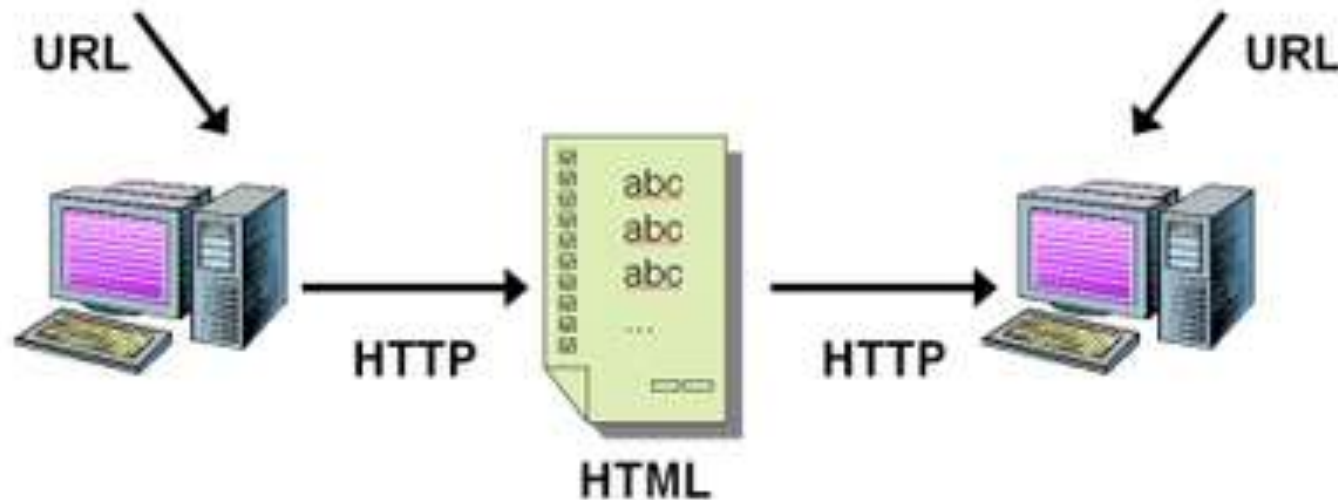
### [Getting code](#)

Getting the code by [anonymous FTP](#), etc.



La Web se basa en tres conceptos fundamentales:

- Los URL
- El protocolo HTTP
- El lenguaje HTML



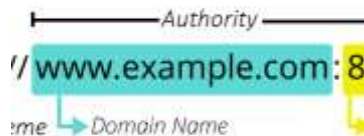
## URL (Uniform Resource Identifier)

El URL sirve para hacer referencia a documentos u objetos alojados en equipos de una red accesibles mediante protocolos existentes.





Un URL está compuesto de varias partes:



- **Protocolo** utilizado para acceder al recurso, en el caso de la Web es HTTP o HTTPS
- **Nombre y dominio del equipo** que contiene el recurso.
- **Puerto** del equipo que contiene el recurso por el que acceder al mismo. Por defecto son los **puertos 80 y 443** para HTTP y HTTPS respectivamente.



Un URL está compuesto de varias partes:

- **Nombre y ruta** del recurso.
- **Parámetros**, utilizados en páginas Web dinámicas para enviar información extra al servidor.
- **Ancla**, para hacer referencia a una ubicación dentro de una página Web.



Es muy importante tener en cuenta la sintaxis de una URL:

- `://` → Para separar el protocolo del nombre de dominio
- `:` → Entre nombre de dominio y puerto
- `/` → Como separador si el recurso no está en el directorio raíz.
- `?` → Para indicar que a continuación van los parámetros.
- `=` → Sirve para asignar un valor a cada parámetro
- `&` → Separador entre parámetros
- `#` → Antes del ancla

## HTTP (Hypertext Transfer Protocol)

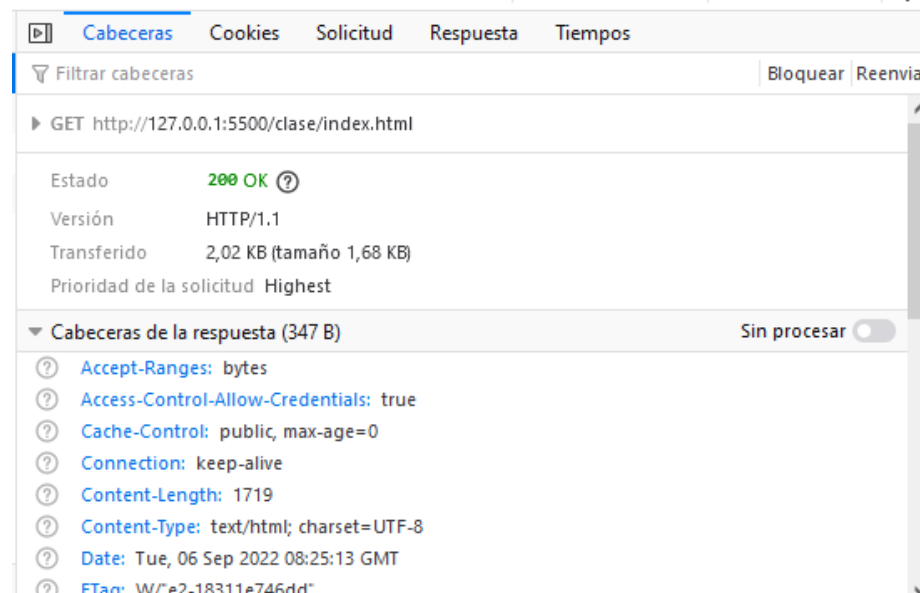
Es un **protocolo de la capa de aplicación** utilizado para la transferencia de documentos hipermedia.

Sirve para establecer las normas que hay que seguir cuando se quiere solicitar una página Web a un servidor.



Ejemplos de elementos definidos en HTTP:

- **Método:** GET para obtener una página Web o POST para enviar datos al servidor (por ejemplo de un formulario).
- **Estado:** para indicar el tipo de respuesta del servidor.





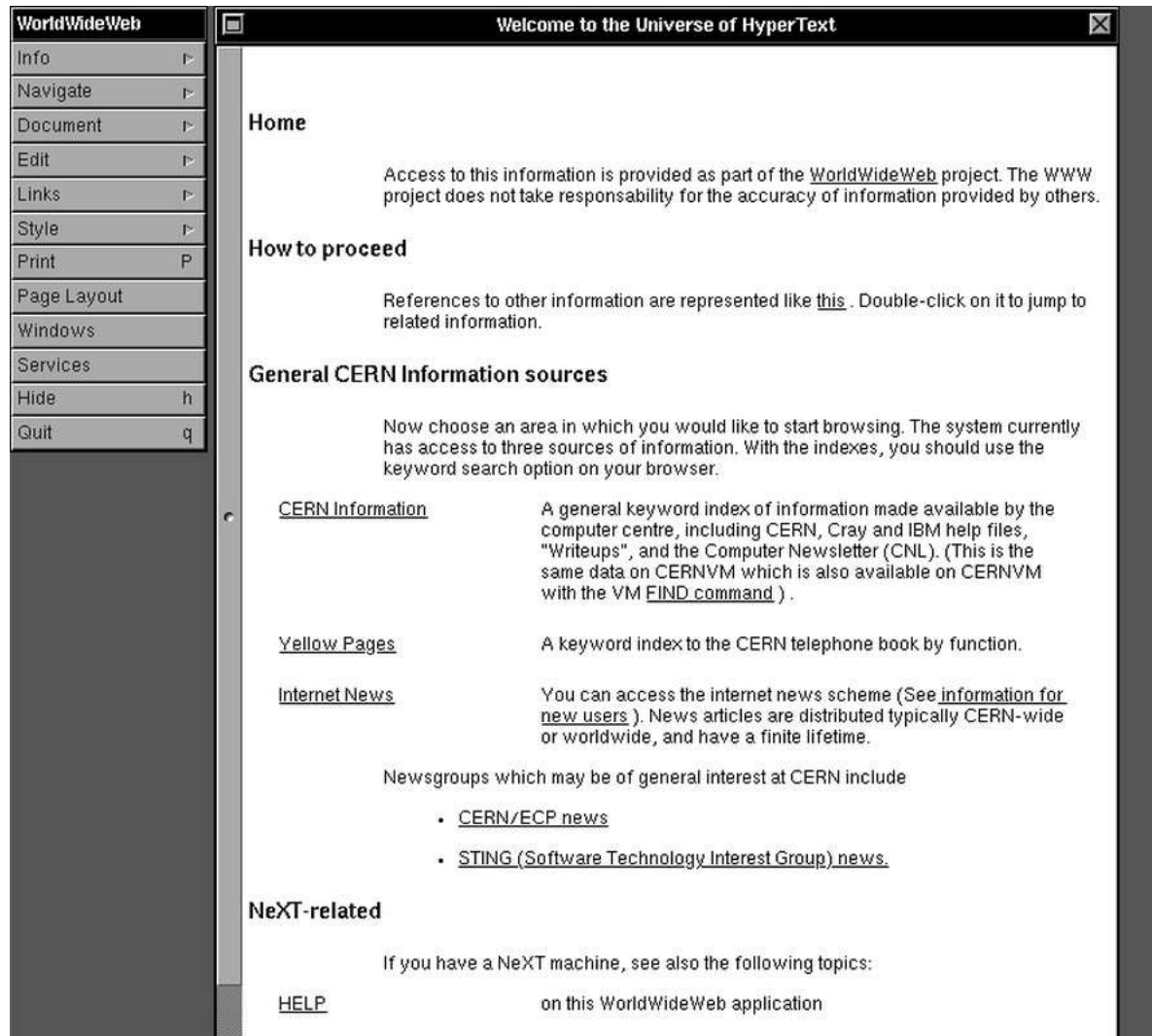
## HTML (Hypertext Markup Language)

Es un **lenguaje de marcas** que define el significado y la estructura del contenido de las páginas Web.

```
7   <body>
8       <h1>Desarrollo de Aplicaciones Web (DAW)</h1>
9       <h2>Listado de módulos</h2>
10      <ul>
11          <li>Desarrollo Web en entorno cliente</li>
12          <li>Desarrollo Web en entorno servidor</li>
13          <li>Diseño de interfaces Web</li>
14          <li>Despliegue de aplicaciones Web</li>
15          <li>Empresa e iniciativa emprendedora</li>
16      </ul>
17  </body>
```

El lenguaje HTML no define la apariencia de cada elemento de la página, sino la **semántica** de dichos elementos.

Para poder acceder a la Web, Tim Bernes-Lee creó un primer servidor web (httpd) y un navegador al que llamó WorldWideWeb.



En 1991 Nicola Pellow, estudiante de matemáticas del CERN desarrolló un navegador Web para terminales en modo texto.

```
Wikipedia – Die freie Enzyklopädie

WIKIPEDIA:HAUPTSEITE

WILLKOMMEN BEI WIKIPEDIA

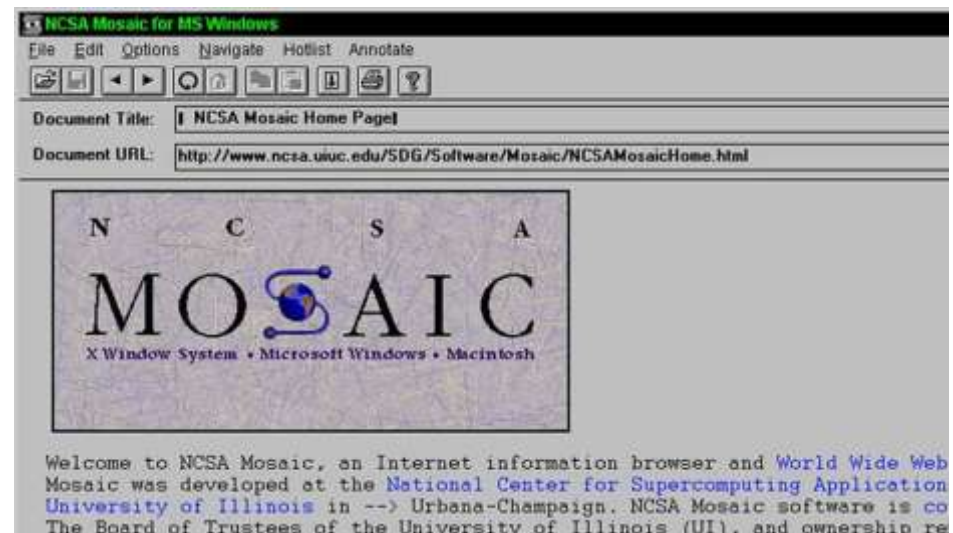
Wikipedia[1] ist ein Projekt zum Aufbau einer Enzyklopädie aus freien
Inhalten[2] in allen Sprachen der Welt. Jeder kann mit seinem Wissen beitragen.
Seit Mai 2001 sind so 1.094.841 Artikel in deutscher Sprache entstanden. Gute
Autorinnen und Autoren sind stets willkommen[3].

[4] Geographie[5]    [6] Geschichte[7]    [8] Gesellschaft[9]    [10] Kunst und
Kultur[11]    [12] Religion[13]    [14] Sport[15]    [16] Technik[17]    [18]
Wissenschaft[19]

1-2, Back, Quit, or Help:
```

En 1993 se desarrolló **Mosaic**, creado en el Centro Nacional de Aplicaciones de Supercomputación (NCSA) de la Universidad de Illinois.

Mosaic se ejecutaba en ordenadores Windows y era fácil de usar, por lo que permitía que cualquier persona pudiera acceder a la Web.

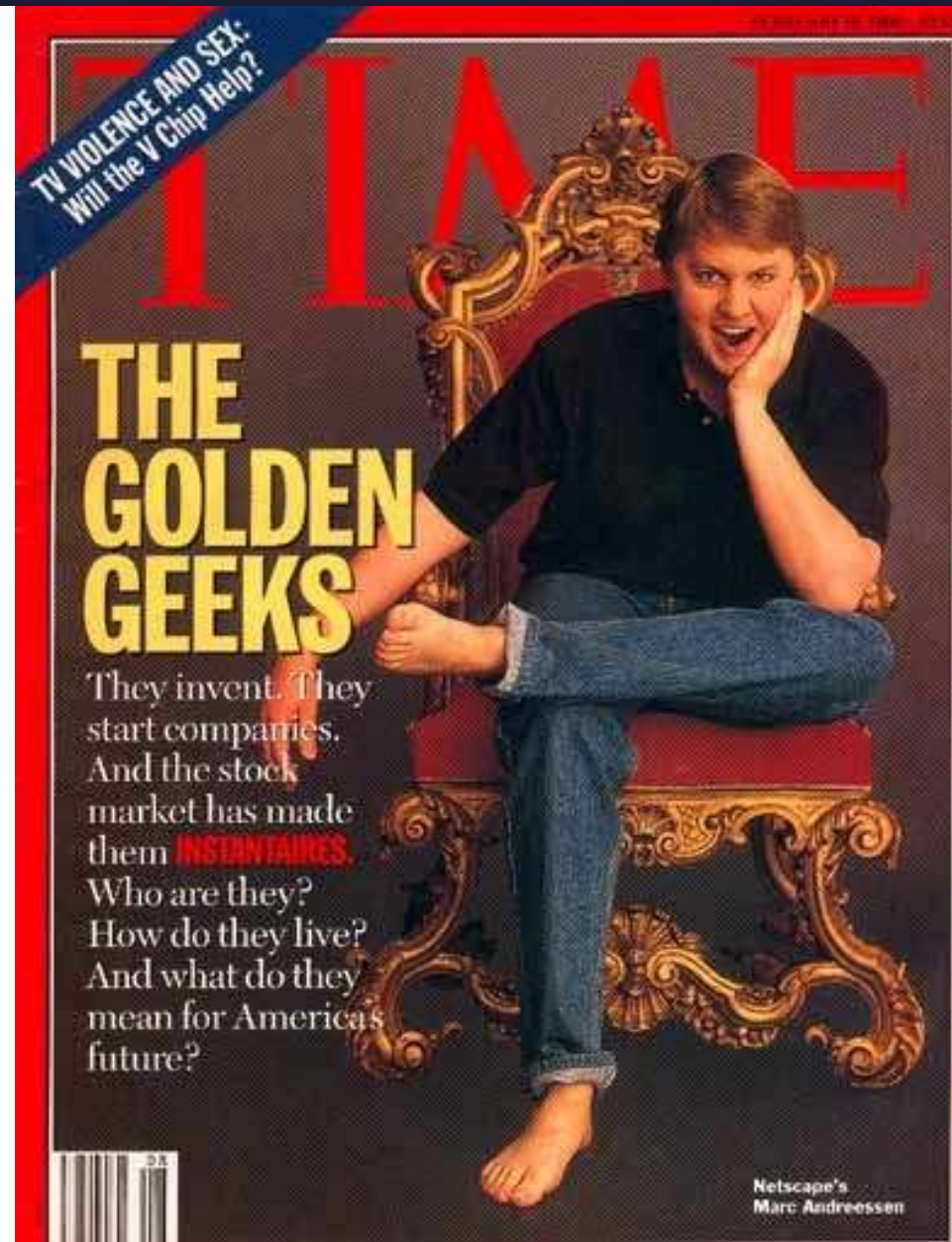


El siguiente año (1994)  
Andreessen fundó  
Netscape y lanzó  
**Netscape Navigator.**

En 1995 Microsoft lanzó  
**Internet Explorer.**



Aquí comenzó la **guerra**  
**de los navegadores.**







En 1995 **Brendan Eich** creó **Javascript** para Netscape Navigator.

La idea era ampliar las limitaciones de la Web, permitiendo que las páginas web ejecutaran scripts en el navegador.

En un principio se llamó **LivesScript**, pero para aprovechar el tirón que tenía Java en ese momento firmó un alianza con Sun Microsystems para llamarlo Javascript.

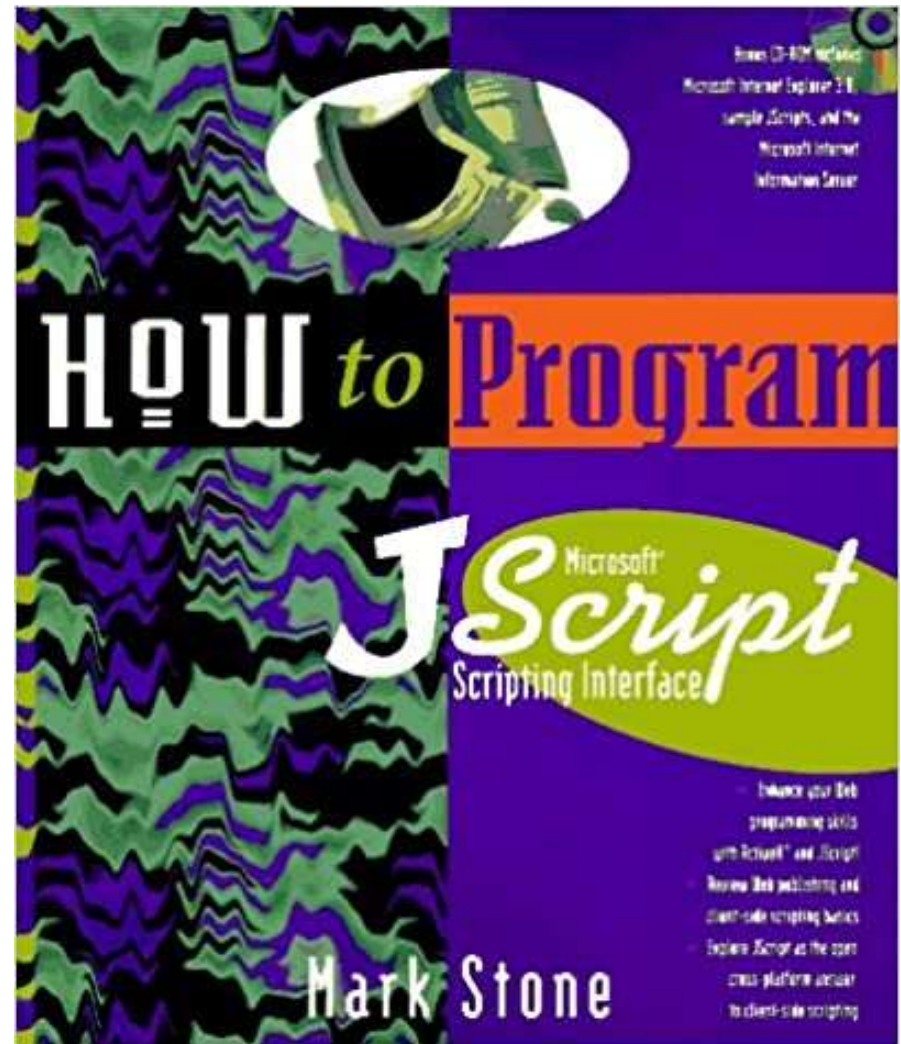
Sin embargo, hay que tener en cuenta que **Javascript no tiene nada que ver con Java**.





Para contratar, Microsoft incluyó soporte (*reducido*) para **CSS (Cascading Style Sheets)** en Internet Explorer 3.0.

También incluyó una versión propia de Javascript, a la que denominó **Jscript**.



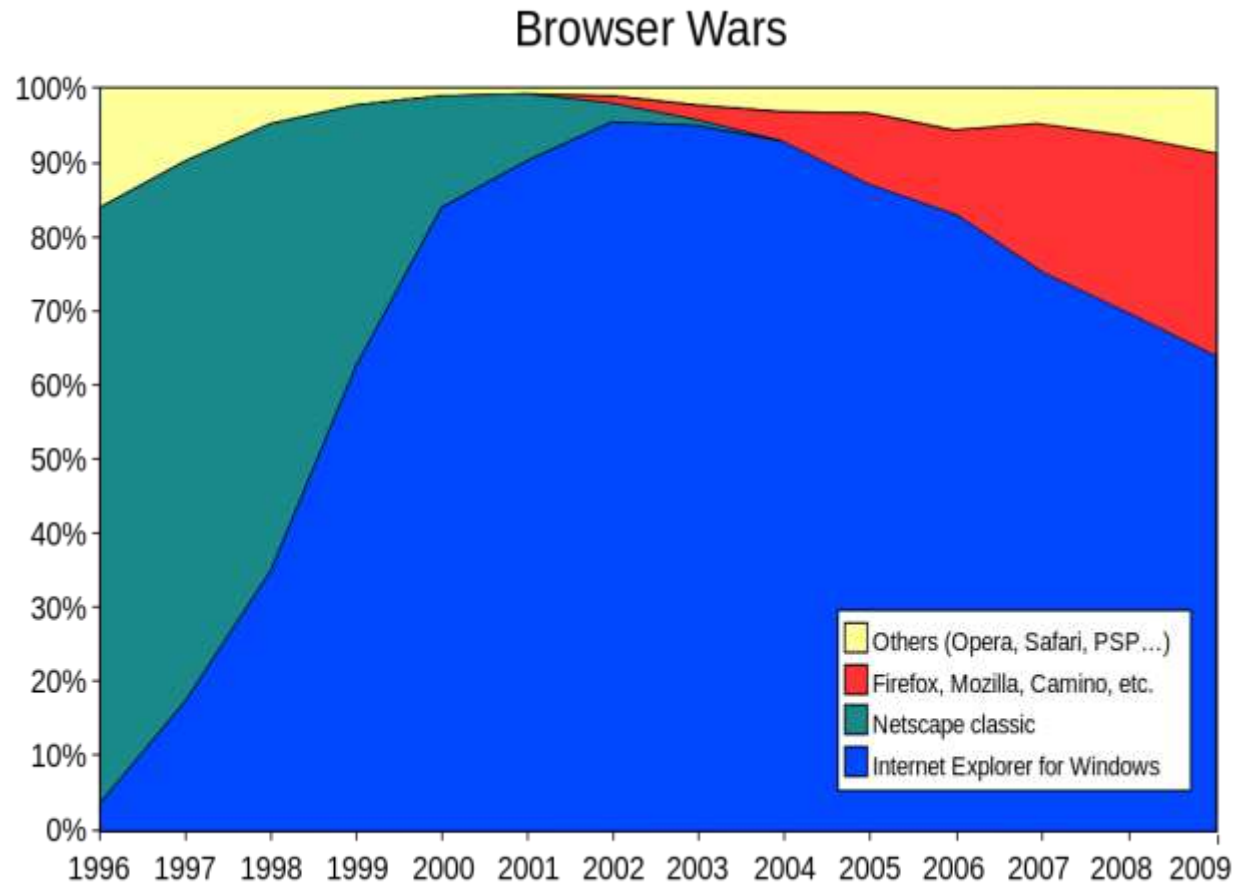
Se puede decir que Microsoft ganó la guerra de los navegadores cuando comenzó a distribuir Internet Explorer con Windows, lo que hizo que en 4 años tuviera el 75% del mercado y para 1999 ya contaba con el 99%.

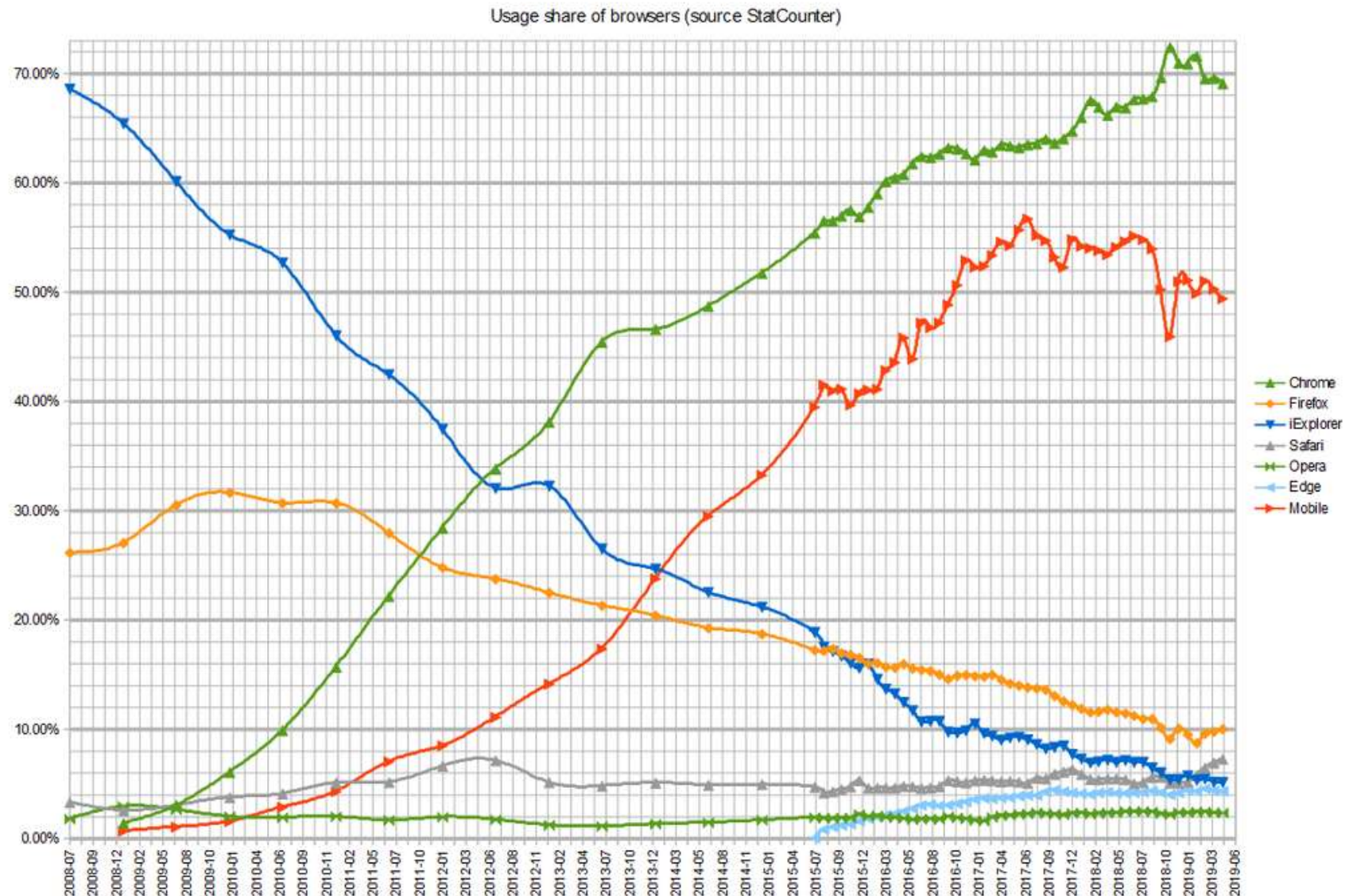
Finalmente, Netscape decidió liberar su código fuente y creó la organización sin ánimo de lucro **Mozilla**.



En 2002, la fundación Mozilla lanzó su navegador de software libre **Firefox**.

Para 2010 la cuota de mercado de Internet Explorer ya se había reducido al 50%.





Volviendo a Javascript, cuando solo estaban iExplorer y Netscape Navigator comenzó a haber problemas por las **diferentes implementaciones** entre Jscript y Javascript.

Por ello, se hizo una propuesta a **ECMA** (*European Computer Manufacturers Association*) para unificarlo en un estándar denominado **ECMAScript**.

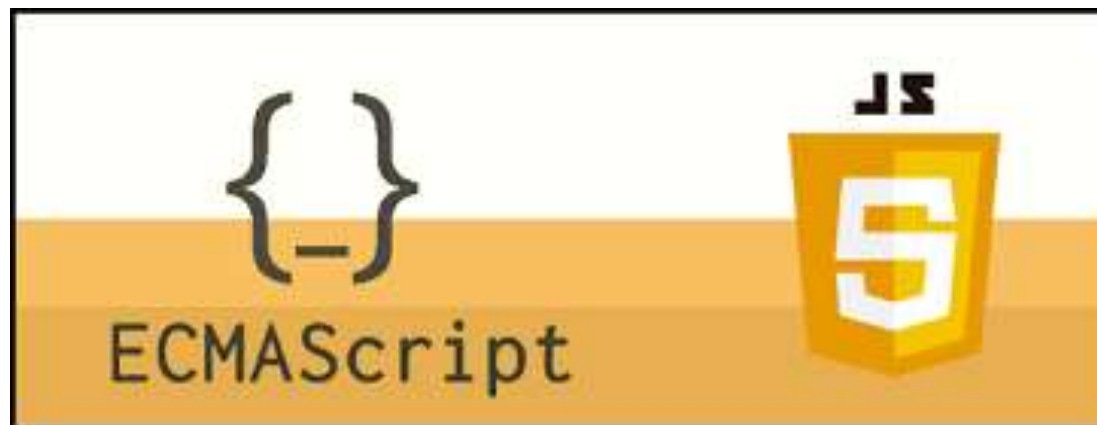




**ECMAScript** es una especificación para crear un lenguaje de scripting de propósito general.

ECMAScript provee las reglas, detalles y pautas que un lenguaje debe cumplir para ser compatible con ECMAScript.

Teniendo esto en cuenta, **Javascript es un lenguaje que se ajusta a la especificación ECMAScript.**



Ha habido diversas versiones de ECMAScript.

### ECMAScript 3 (1999)

- Soporte de expresiones regulares
- Nuevas sentencias de control
- Manejo de excepciones
- Mejoras en el manejo de Strings



## ECMAScript 5 (2011)

- Getters y setters
- Introducción de métodos estáticos de Object
- Curricación de funciones
- Cambios en el objeto Date
- Soporte nativo de JSON

## ECMAScript 6 (2015)

- Módulos
- Ámbito a nivel de bloque (sentencia let)
- Desestructuración
- Parámetros rest y por defecto
- Iteradores
- Comprensión de arrays
- Tablas hash y mapas weak

Aunque actualmente todos los navegadores tienen un buen soporte de Javascript, hay características que pueden no estar implementadas según la versión del navegador.

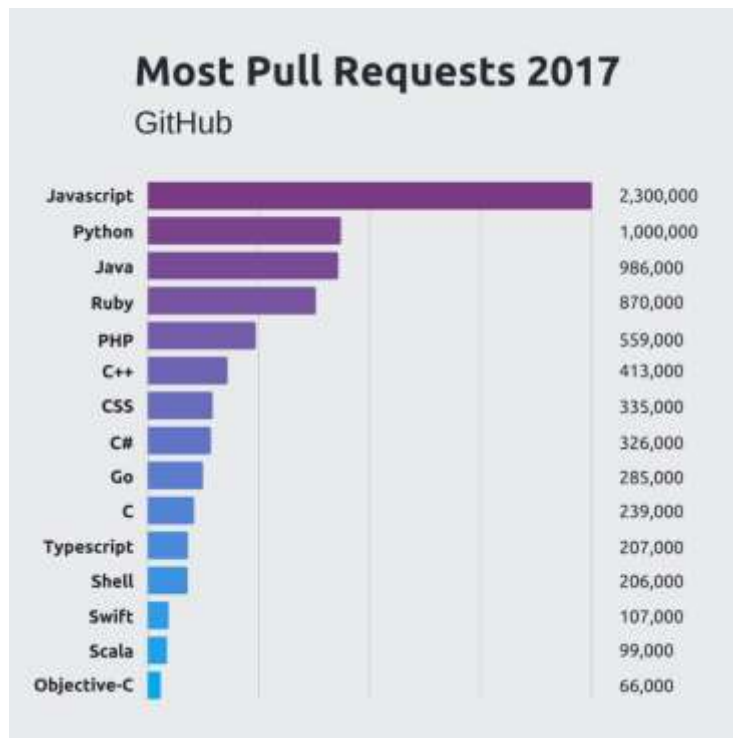
Podemos consultarlo en la web <https://caniuse.com/> que refleja el soporte de ES6 en los diferentes navegadores.



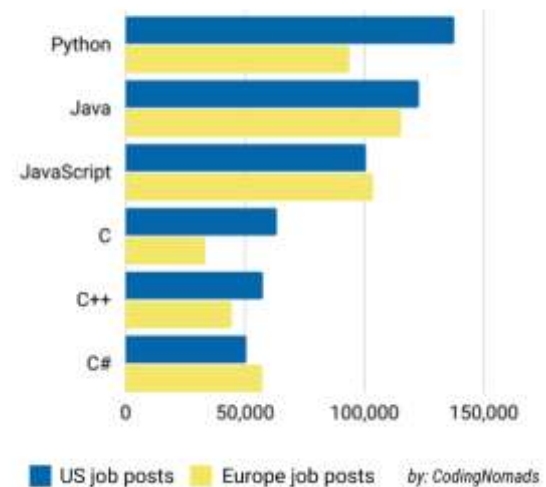
0 en <https://kangax.github.io/compat-table/es6/>

[illegible]

¿Dónde está JavaScript en la actualidad? Es difícil de determinar algo tan difuso como la popularidad de un lenguaje de programación ya que hay muchas formas de hacerlo, aunque la mayoría de indicadores coinciden en su relevancia.

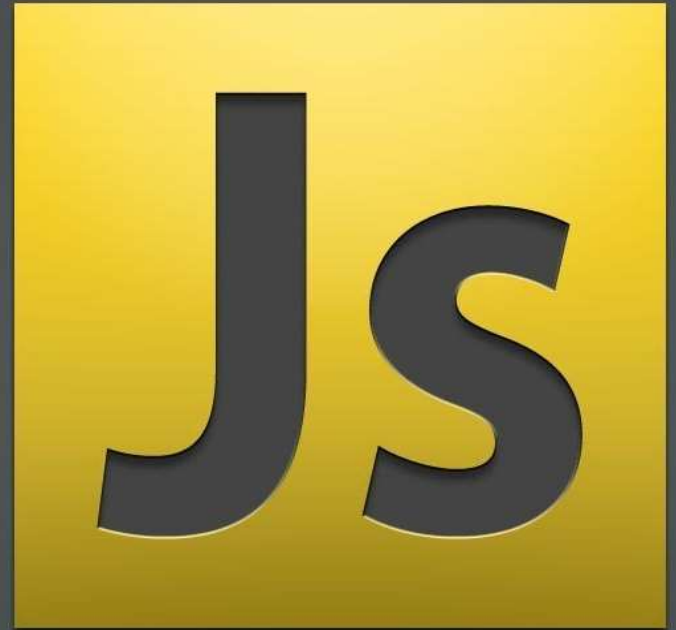


**Most in-demand programming languages 2021-2022**



# 2

## CARACTERÍSTICAS DE JAVASCRIPT



Javascript ha evolucionado notablemente desde sus inicios, pudiendo considerarse en la actualidad, como ES6, un lenguaje totalmente maduro.

Algunas **características** que podemos destacar son:

- Interpretado
- Lenguaje del lado del cliente
- Disponibilidad de frameworks
- Lenguaje de tipado débil
- Multi-paradigma



## Lenguaje interpretado

Tradicionalmente los lenguajes de programación se han dividido en dos grupos:

- **Lenguajes compilados:** el código fuente del programa es ejecutado en un compilador que lo convierte a **código máquina (ensamblador)**, generando un archivo **ejecutable**.
- **Lenguajes interpretados:** el código fuente es leído y ejecutado línea a línea, sin generar ningún archivo intermedio.

Compilados		Interpretados	
PROS	CONS	PROS	CONS
Solo requiere ejecutable para ejecutarse	No es multiplataforma	Multiplataforma	Se requiere intérprete
Más rápido	Inflexible	Pruebas más sencillas	Más lento
Código fuente privado	Paso extra	Más fácil de depurar	Código fuente es público

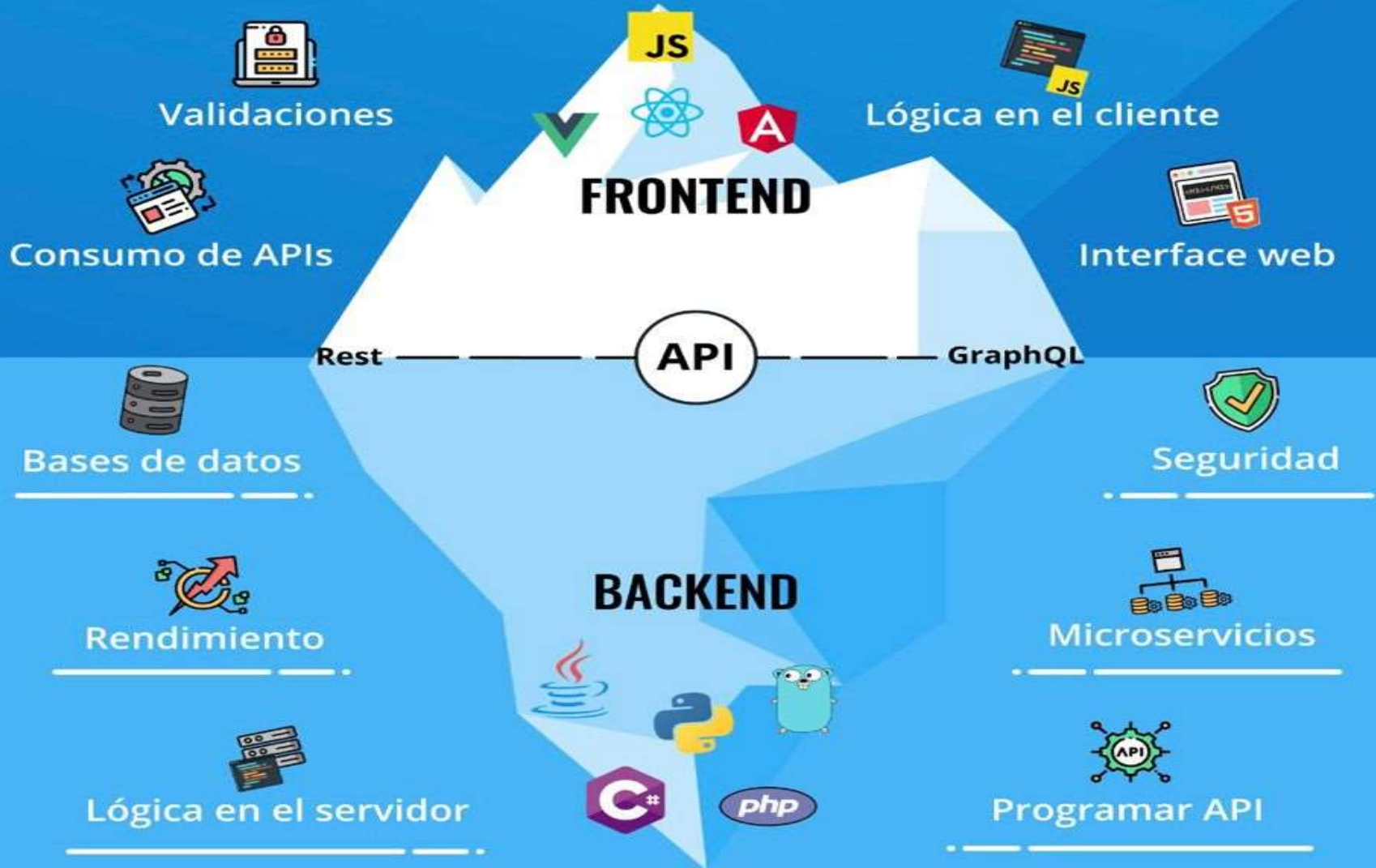
## Lenguaje del lado cliente

El desarrollo web se divide en dos áreas: frontend y backend.

- El **backend** hace referencia a la programación en el lado del servidor. Hay múltiples lenguajes de programación en el backend, como PHP, Python, Ruby, ASP o incluso el propio Javascript (Node.js)
- El **frontend** es la programación en el lado del cliente y Javascript es el único lenguaje de programación utilizado.

Normalmente se denomina **fullstack** al desarrollador que programa en ambas áreas.

# ¿QUÉ ES BACKEND Y FRONTEND?



## Frameworks

Un **framework** es una estructura sobre la que desarrollar software orientado a una tarea particular.

Normalmente proporcionan librerías y un armazón sobre el que empezar a desarrollar simplificando así la realización de determinadas tareas.

Javascript dispone de múltiples frameworks.



**jQuery** es una librería desarrollada en 2006 cuyo objetivo era simplificar el desarrollo con Javascript.

Permite manipular fácilmente el DOM de las páginas web.

También simplifica la realización de consultas **AJAX** (***Asynchronous Javascript and XML***), que permiten que una webapp mantenga una comunicación asíncrona con el servidor en segundo plano.





**Angular JS** es un framework de código abierto desarrollado por Google.

Es utilizado para crear **Single Page Applications (SPA)**, aplicaciones que consisten en una única página que se va actualizando dinámicamente en el lado del cliente.



**React** ha sido diseñado por Facebook y, al igual que Angular, sirve para crear aplicaciones SPA.

Está basado en componentes donde cada uno tiene su propia interfaz (programada en JSX) y su lógica y estado.

Permite crear aplicaciones nativas para Android con React Native.



**Vue.js** es otro framework basado en componentes y que se caracteriza por su simplicidad.

Se suele utilizar para el frontend de las aplicaciones desarrolladas con **Laravel** (un framework de PHP)



No todos los frameworks son para programar en el lado cliente, también hay muchos para Node.js.

**Express** proporciona mecanismos para tareas como:

- Generar vistas basadas en plantillas
- Manejar peticiones HTTP
- Añadir procesamiento de peticiones **middleware**

## De tipado débil o no tipado

Un **lenguaje de tipado débil** es aquel en el cual las variables no están limitadas a un único tipo de datos.

También se denomina de **tipado dinámico**, ya que el intérprete asigna el tipo a las variables en tiempo de ejecución en función de su valor.

Permite más libertad al programar, pero puede provocar que se comentan más error.

Lo opuesto son **lenguajes de tipado fuerte** como C++ o Java.

## Multi-paradigma

Javascript se puede adaptar a estilos de programación o **paradigmas**:

- **Imperativo**: los programas se pueden definir como una serie de órdenes que se ejecutan de forma secuencial. Otros lenguajes son C, C++, Fortran, ...
- **Orientado a objetos**: Javascript soporta elementos clave de la orientación a objetos, como el polimorfismo, la encapsulación o la herencia. Otros lenguajes que soportan este paradigma son C++, C# o Java

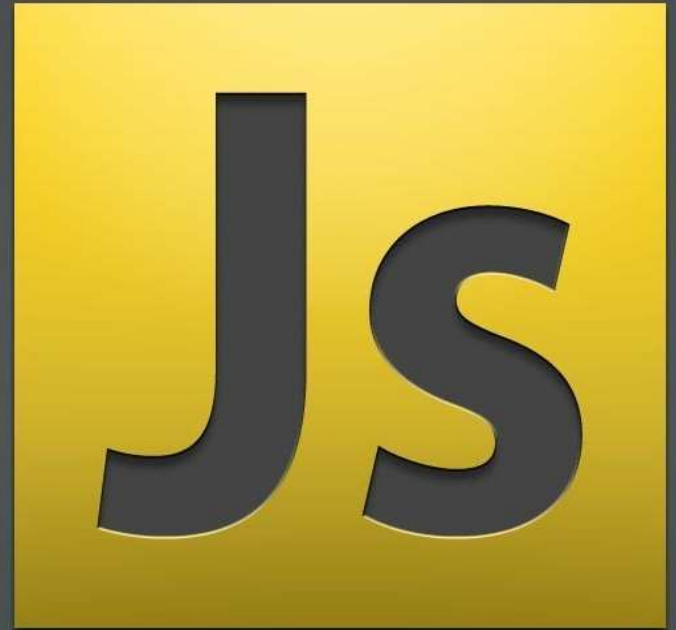


- **Funcional:** En Javascript las funciones son *objetos de primera clase*, lo que quiere decir que las funciones se pueden tratar como cualquier otro valor del lenguaje (almacenarlas en variables, pasarlas como parámetro o devolverlas desde funciones). Esto permite programar en Javascript programar en Javascript utilizando un enfoque funcional.

<https://opensource.com/article/17/6/functional-javascript>

# 3

## PREPARACIÓN DEL ENTORNO



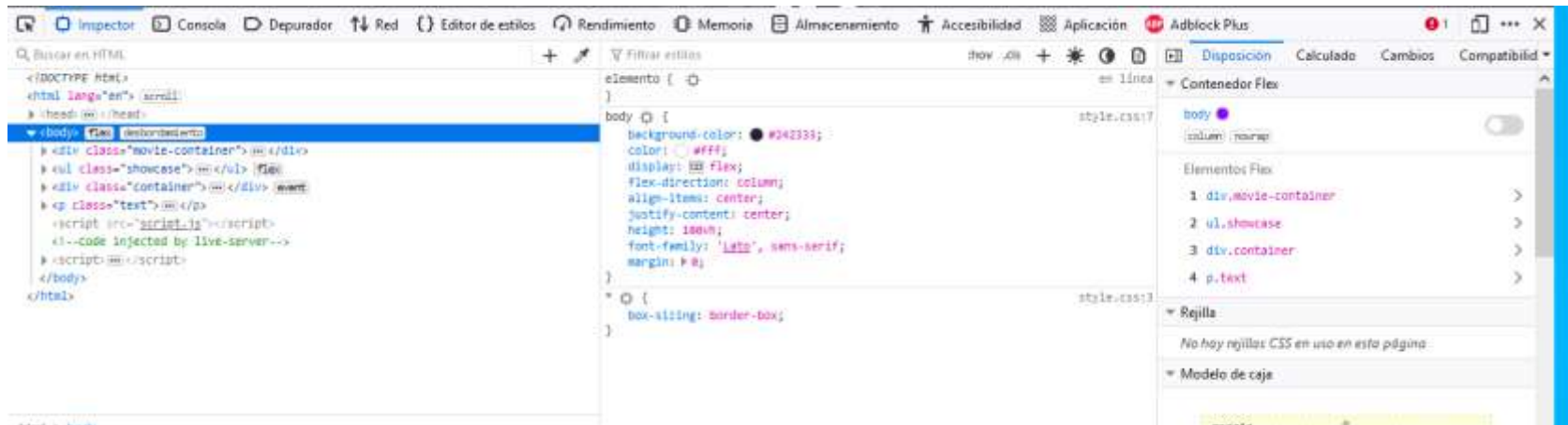
Software que vamos a instalar:

- Navegador: **Mozilla Firefox**
- Editor: **Visual Studio Code**
- Sistema de control de versiones: **Git**
- Entorno de ejecución: **Node.js**



## Navegador: Mozilla Firefox

Para la prueba, ejecución y depuración de las páginas que realicemos utilizaremos el navegador Mozilla Firefox.



Utilizaremos las herramientas del desarrollador, accesible mediante Ctrl + Shift + i

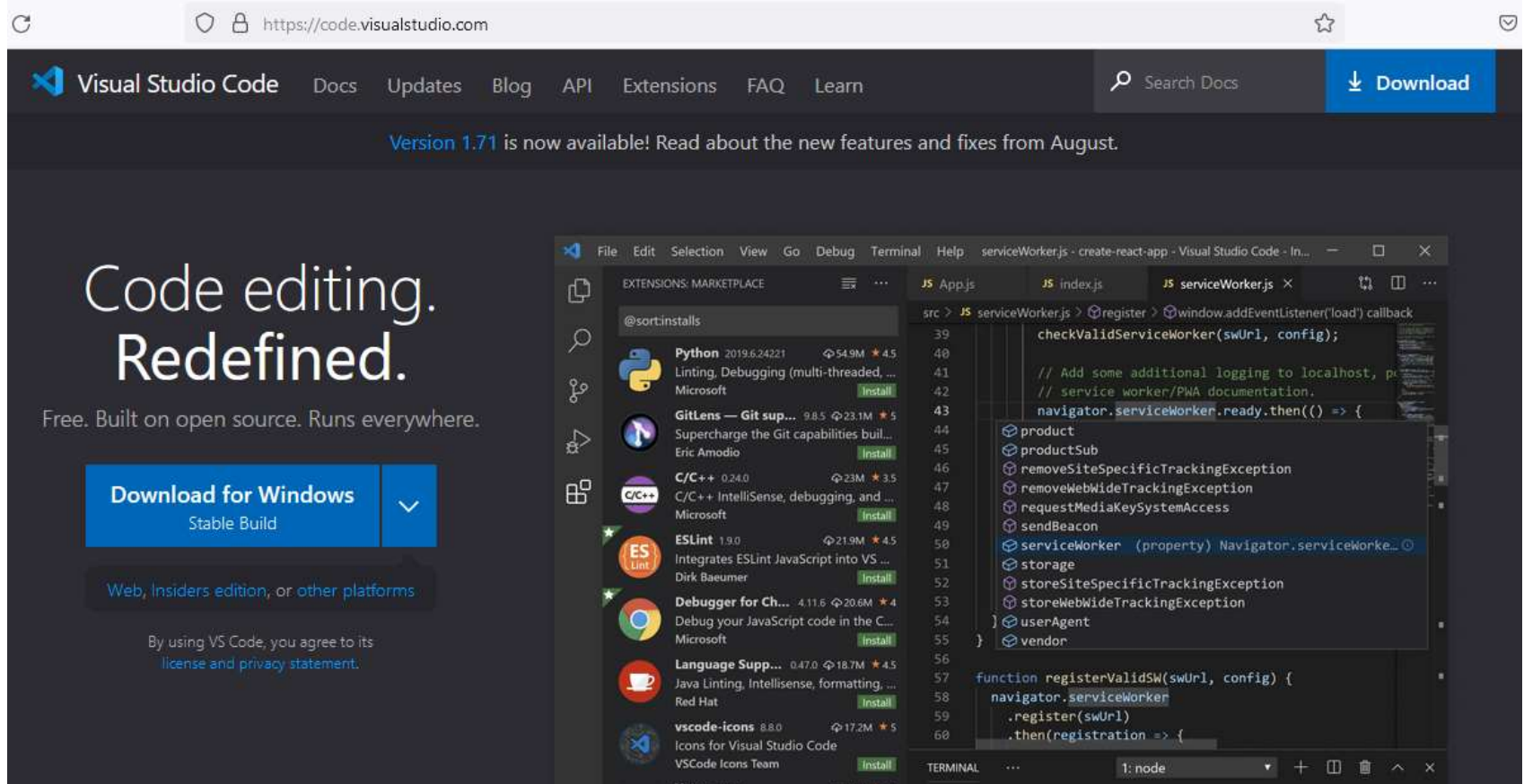
También puedes instalar la versión **Firefox Browser Developer**, con herramientas específicas para desarrolladores.



## Welcome to Firefox Browser Developer Edition


Firefox has been rebuilt from the ground-up to be  
faster, sleeker, and more powerful than ever.

## Editor: Visual Studio Code





Personalizamos el entorno de trabajo: iconos y tema de color.




### Material Icon Theme v4.20.0

Philipp Kief | 13.813.655 | ★★★★★ (263) | ❤️ Patrocinador

Material Design Icons for Visual Studio Code

[Establecer tema para iconos de archivo](#) [Deshabilitar](#) [Desinstalar](#) ⚙️

Esta extensión está habilitada globalmente.



### Bluloco Dark Theme v3.7.2

Umut Topuzoğlu | 180.906 | ★★★★★ (28)

A fancy but yet sophisticated dark designer color scheme / theme.

[Configurar tema de color](#) [Deshabilitar](#) [Desinstalar](#) ⚙️

Esta extensión está habilitada globalmente.

Los **snippets** son palabras clave que generan fragmentos de código, optimizando el tiempo de trabajo de codificación.



### JavaScript (ES6) code snippets v1.8.0

charalampos karypidis | 9,374,915 | ★★★★★ (33)

Code snippets for JavaScript in ES6 syntax

Instalar



### ES7+ React/Redux/React-Native snippets v4.4.3

dsznajder | 6,046,490 | ★★★★★☆ (61)

Extensions for React, React-Native and Redux in JS/TS with ES7+ syntax. Customizable. Built-in integration with prettier.

Deshabilitar ▼

Desinstalar ▼

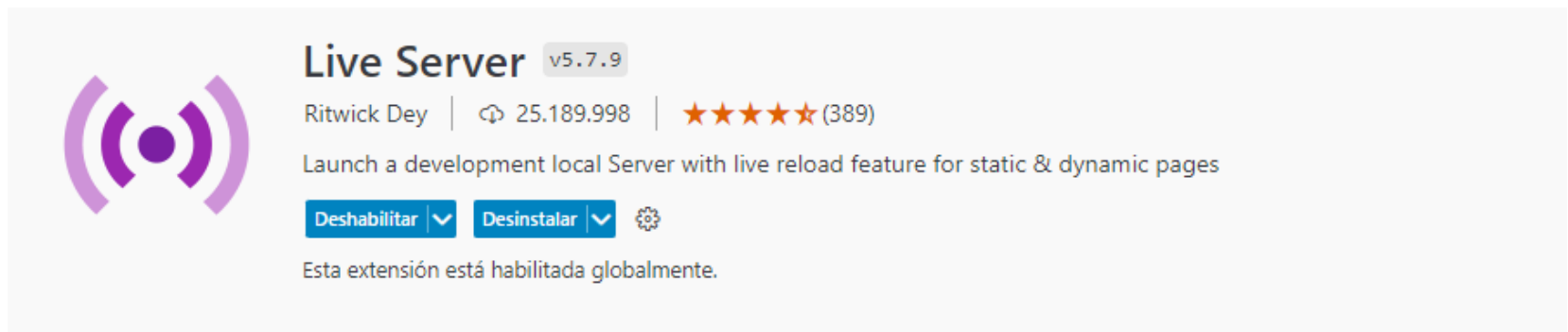


Esta extensión está habilitada globalmente.

Snippet	Contenido
fre →	forEach loop in ES6 syntax <code>array.forEach(currentItem =&gt; {})</code>
fof →	for ... of loop <code>for(const item of object) {}</code>
fin→	for ... in loop <code>for(const item in object) {}</code>
anfn→	creates an anonymous function <code>(params) =&gt; {}</code>
nfn→	creates a named function <code>const add = (params) =&gt; {}</code>
dob→	destructing object syntax <code>const {rename} = fs</code>
dar→	destructing array syntax <code>const [first, second] = [1,2]</code>
sti→	set interval helper method <code>setInterval(() =&gt; {});</code>
sto→	set timeout helper method <code>setTimeout(() =&gt; {});</code>
prom→	creates a new Promise return <code>new Promise((resolve, reject) =&gt; {});</code>

Listado completo en <https://marketplace.visualstudio.com/items?itemName=xabikos.JavaScriptSnippets>

Para ver como cambian nuestras páginas a medida que escribimos códigos utilizaremos **Liver Server**.



## Sistema de control de versiones: Git

Un sistema de control de versiones es una aplicación software que permite **gestionar los cambios en el código fuente** a lo largo del tiempo.

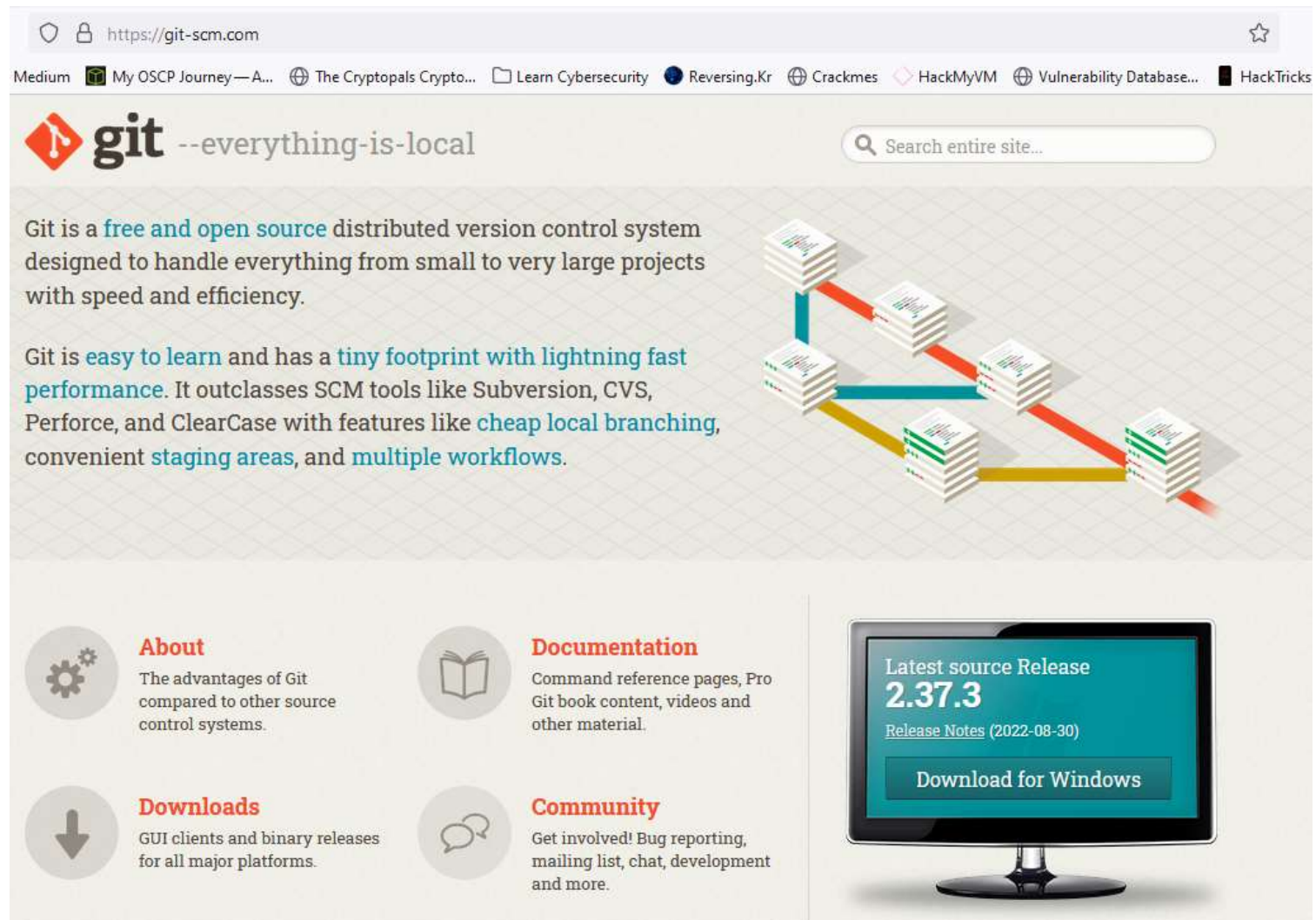


Git es un sistema de control de versiones distribuido, lo que quiere decir que **no está centralizado**.

### **Características:**

- No depende de un repositorio central.
- Es software libre
- Tiene un sistema de trabajo con ramas
- Las ramas permiten crear proyectos divergentes de un proyecto principal, para experimentar o probar nuevas funcionalidades.





The screenshot shows the Git website homepage. At the top, the browser address bar displays <https://git-scm.com>. Below the address bar, a navigation bar contains links to various resources: Medium, My OSCP Journey—A..., The Cryptopals Crypto..., Learn Cybersecurity, Reversing.Kr, Crackmes, HackMyVM, Vulnerability Database..., and HackTricks. The main header features the Git logo (a red octocat) and the tagline "--everything-is-local". A search bar with the placeholder text "Search entire site..." is located to the right of the header. The main content area describes Git as a "free and open source distributed version control system" designed to handle projects of all sizes with speed and efficiency. It also highlights that Git is "easy to learn" and has a "tiny footprint with lightning fast performance", outclassing tools like Subversion, CVS, Perforce, and ClearCase. The text mentions features like "cheap local branching", "convenient staging areas", and "multiple workflows". To the right of the text is a diagram illustrating Git's distributed version control system, showing multiple repositories (represented as stacks of books) connected by a network of colored lines (red, blue, yellow). Below the main content, there are four sections with icons and text: "About" (gear icon) describing the advantages of Git, "Documentation" (book icon) listing command reference pages and other materials, "Downloads" (download arrow icon) mentioning GUI clients and binary releases, and "Community" (speech bubble icon) encouraging bug reporting and participation. On the right side of the page, there is a large monitor displaying the "Latest source Release 2.37.3" and a button to "Download for Windows".

<https://git-scm.com>

Medium My OSCP Journey—A... The Cryptopals Crypto... Learn Cybersecurity Reversing.Kr Crackmes HackMyVM Vulnerability Database... HackTricks

**git** --everything-is-local

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

**About**  
The advantages of Git compared to other source control systems.

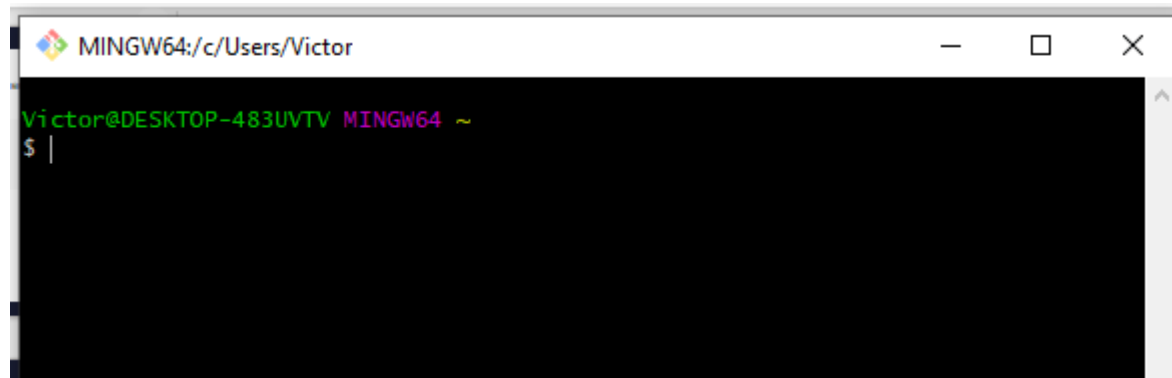
**Documentation**  
Command reference pages, Pro Git book content, videos and other material.

**Downloads**  
GUI clients and binary releases for all major platforms.

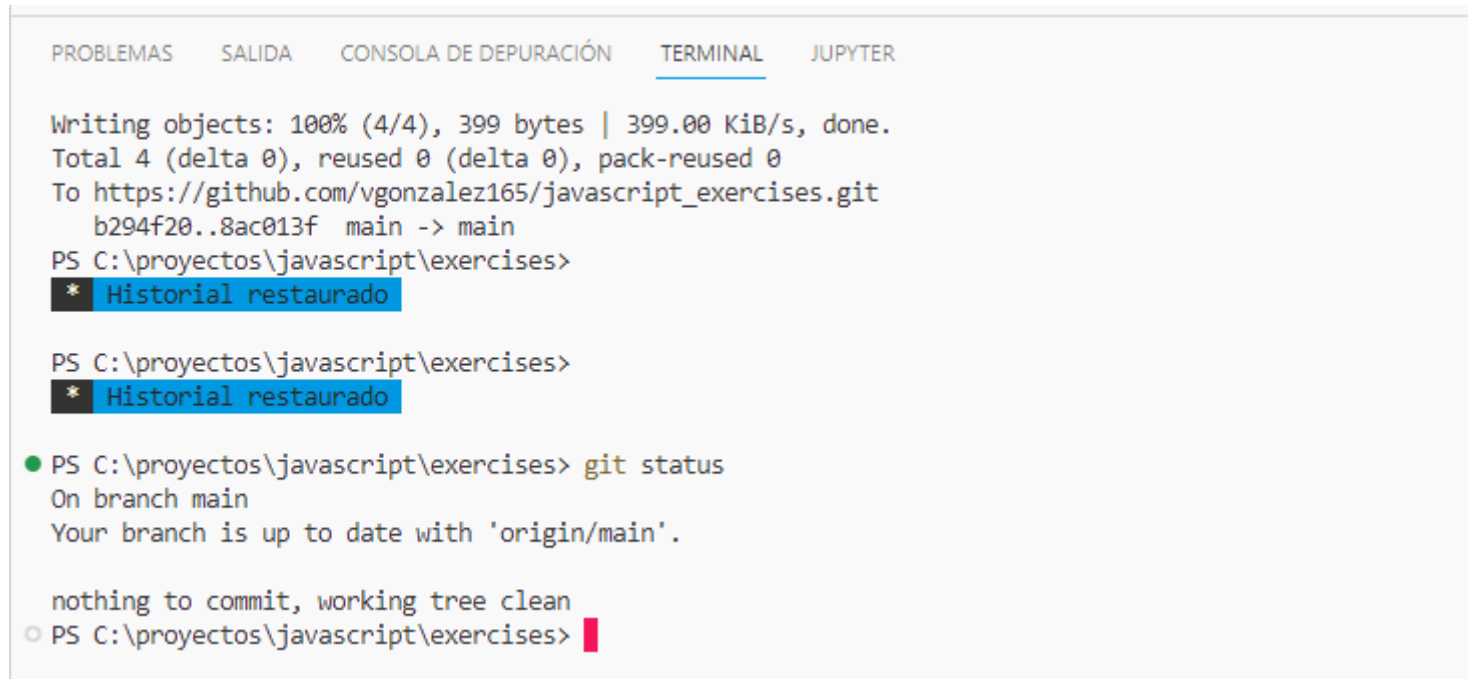
**Community**  
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release  
**2.37.3**  
[Release Notes \(2022-08-30\)](#)  
[Download for Windows](#)

Git incluye su propia terminal, denominada **Git Bash**, aunque se puede ejecutar desde cualquier terminal de Windows (Powershell, cmd, ...)



Normalmente usaremos la terminal integrada de VS Code, que se puede invocar con la combinación `Ctrl+Shift+ñ`



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  JUPYTER

Writing objects: 100% (4/4), 399 bytes | 399.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/vgonzalez165/javascript_exercises.git
   b294f20..8ac013f  main -> main
PS C:\proyectos\javascript\exercises>
* Historial restaurado

PS C:\proyectos\javascript\exercises>
* Historial restaurado

● PS C:\proyectos\javascript\exercises> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
○ PS C:\proyectos\javascript\exercises>
```

También necesitamos un servidor Git. Hay diferentes opciones, tanto gratuitas como de pago. Nosotros usaremos GitHub.



[Product](#) [Team](#) [Enterprise](#) [Explore](#) [Marketplace](#) [Pricing](#)[Sign in](#)[Sign up](#)

# Let's build from here, automatically

The complete developer platform to build,  
scale, and deliver secure software.

[Sign up for GitHub](#)

**83+ million**  
Developers

**4+ million**  
Organizations

**200+ million**  
Repositories

**90%**  
Fortune 100

## Entorno de ejecución: Node.js

**Node.js** es un entorno de ejecución de JavaScript orientado a eventos asíncronos diseñado para crear aplicaciones de red escalables.



Normalmente es utilizado para desarrollar aplicaciones en entorno servidor utilizando JavaScript.

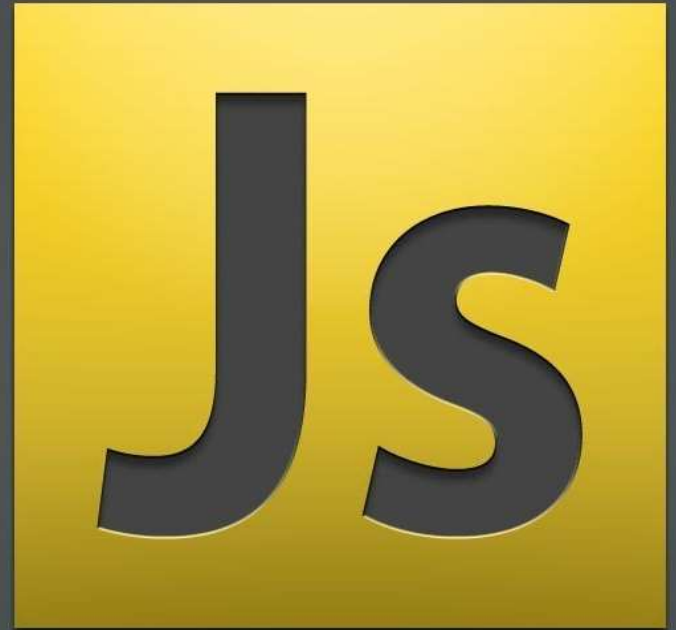
Utilizaremos node.js en la segunda parte del curso, pero lo dejamos ya instalado.





# 4

## EJECUCIÓN DE JAVASCRIPT



Para ejecutar Javascript necesitamos un **motor de Javascript**.

Los más conocidos son:

- **SpiderMonkey**: Firefox
- **V8**: navegadores basados en Chromium (Chrome, MS Edge y Opera) y en Node.js y Deno.
- **JavascriptCore**: navegadores basados en Webkit (Safari)
- **Carakan**: versiones antiguas de Opera
- **Chakra**: intérprete de Jscript de Internet Explorer.

<https://tech.tribalyte.eu/blog-motor-de-javascript>

Por tanto, salvo que lo hagamos en el entorno de ejecución Node.js, necesitaremos un navegador para ejecutarlo.

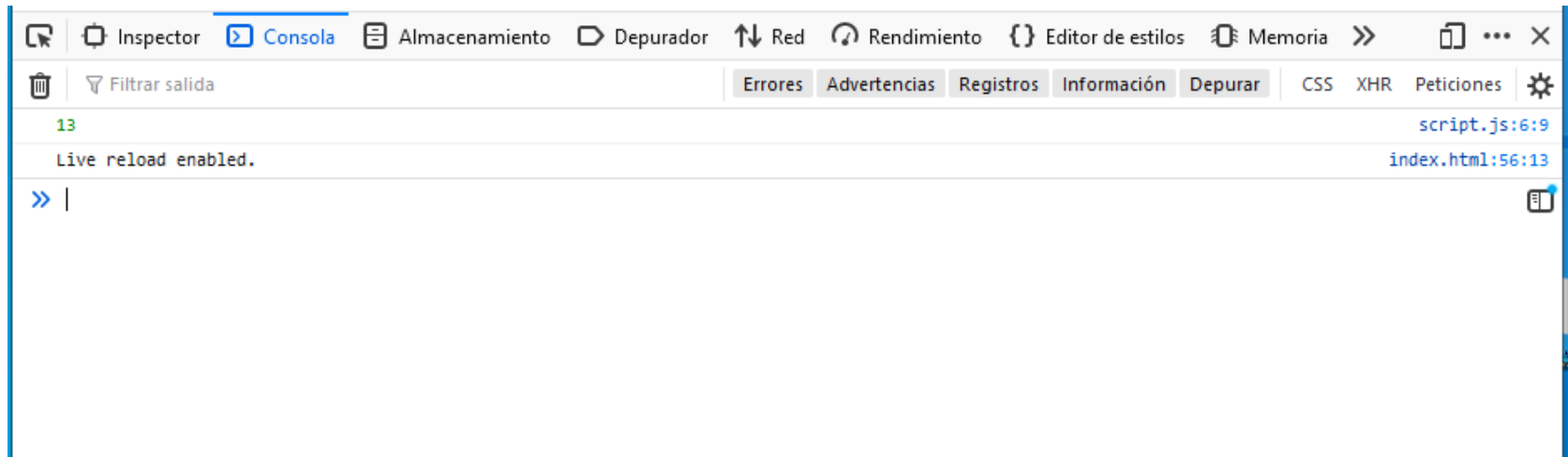
Las opciones que hay son:

- Ejecutarlo directamente en la consola
- Incrustado en un documento HTML
- Mediante un script externo al HTML
- En el entorno de ejecución Node.js

## Ejecución en la consola

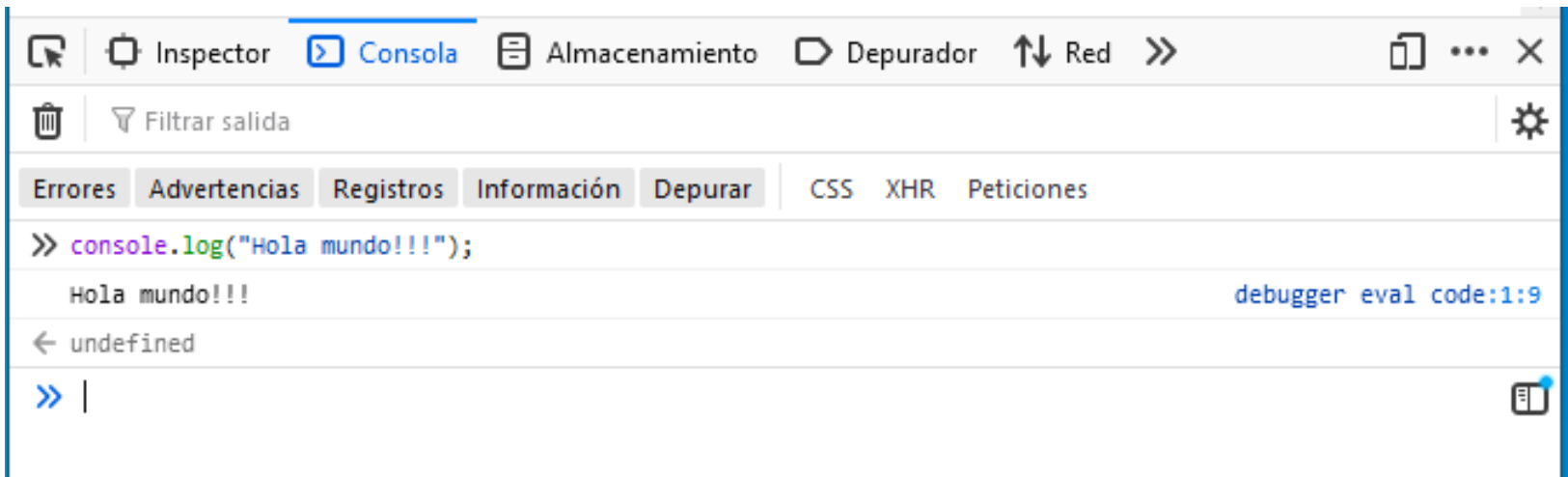
Todos los navegadores tienen una consola a la que se puede acceder desde *Herramientas para desarrolladores*.

En Firefox se puede abrir con la combinación de teclas **Ctrl+Shift+i**



En la consola podemos introducir órdenes pero también es donde se mostrará la salida que enviemos a la consola en nuestros scripts.

Se envía la salida a la consola con la orden `console.log()`



## Incrustado en un documento HTML

En ese caso hay que introducir el código dentro de la etiqueta `<script>`.

```
index.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>CodeWars</title>
8  </head>
9  <body>
10     <h1>Pruebas</h1>
11
12     <script>
13         console.log("Hola mundo!!!");
14     </script>
15 </body>
16 </html>
```

## En un script externo

Se debe utilizar el atributo `src` de la etiqueta `<script>` para referenciar el fichero del script (extensión `.js`)

```
index.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>CodeWars</title>
8  </head>
9  <body>
10     <h1>Pruebas</h1>
11
12     <script src="script.js"></script>
13 </body>
14 </html>
```

## En el entorno de ejecución Node.js

Para ello debemos utilizar el comando `node` desde la línea de comandos pasando el nombre del script como argumento.

```
● PS C:\proyectos\javascript\exercises> dir
```

```
Directory: C:\proyectos\javascript\exercises
```

Mode	LastWriteTime	Length	Name
d----	07/09/2022 17:23		code_wars
d----	07/09/2022 7:44		retos_de_programacion
d----	05/09/2022 22:14		vanilla_web_projects
-a---	07/09/2022 18:13	331	index.html
-a---	07/09/2022 18:12	29	script.js

```
● PS C:\proyectos\javascript\exercises> node script.js
```

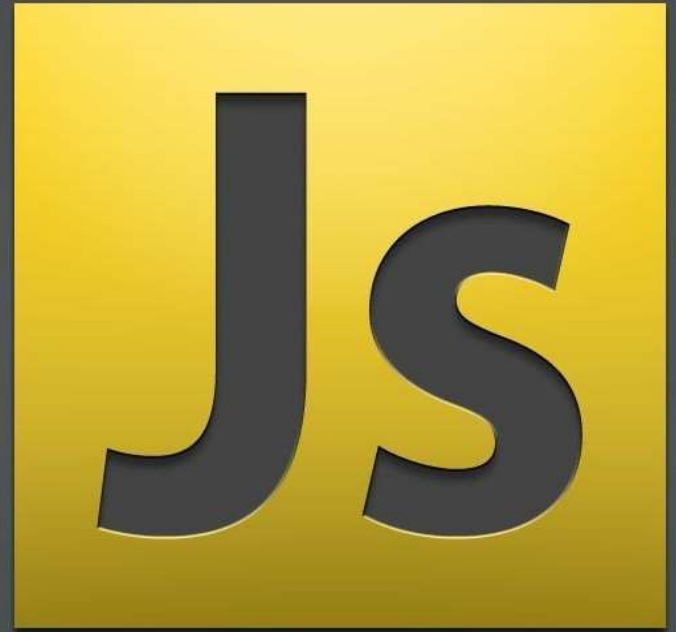
```
Hola mundo!!!
```

```
○ PS C:\proyectos\javascript\exercises> █
```

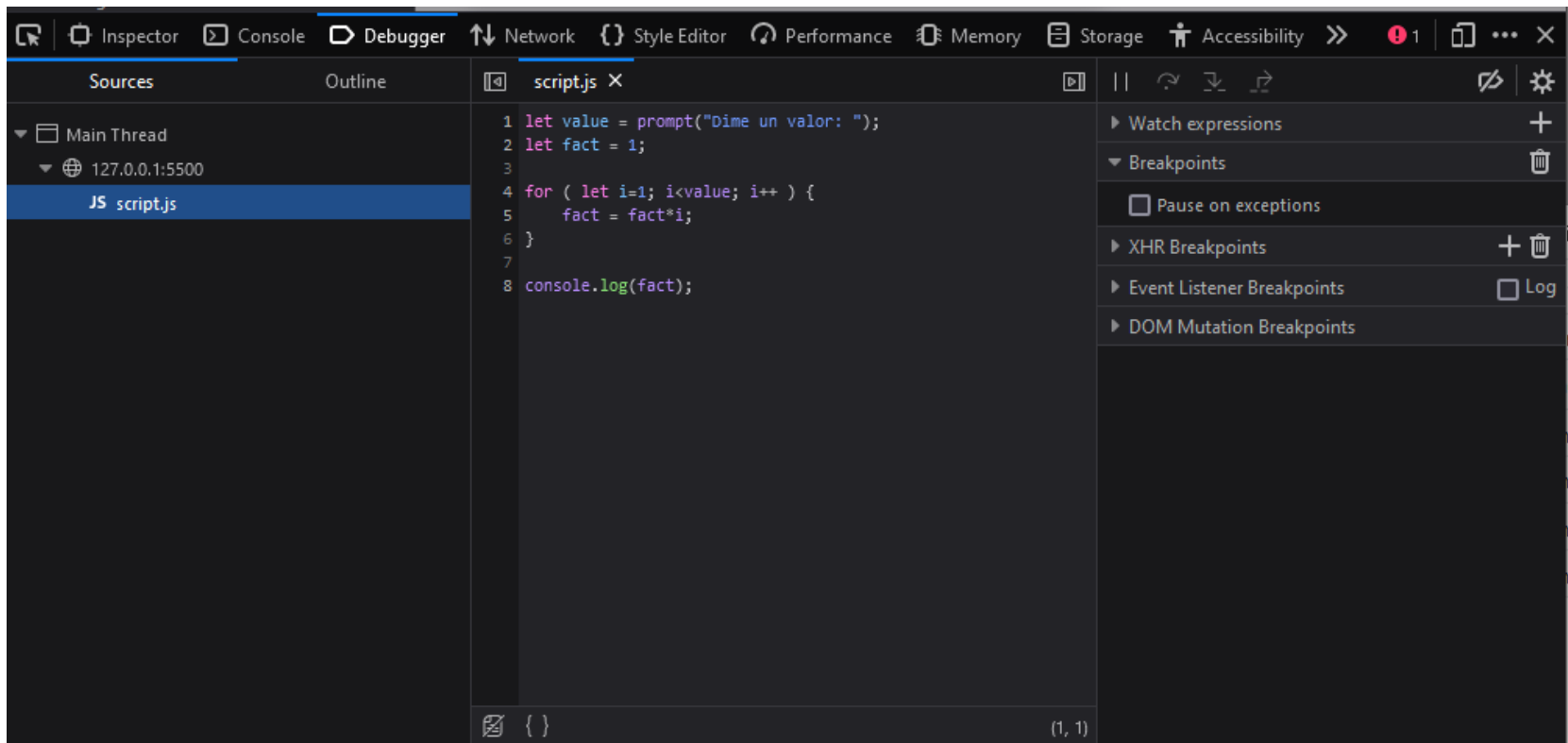


# 5

## DEPURACIÓN DE CÓDIGO



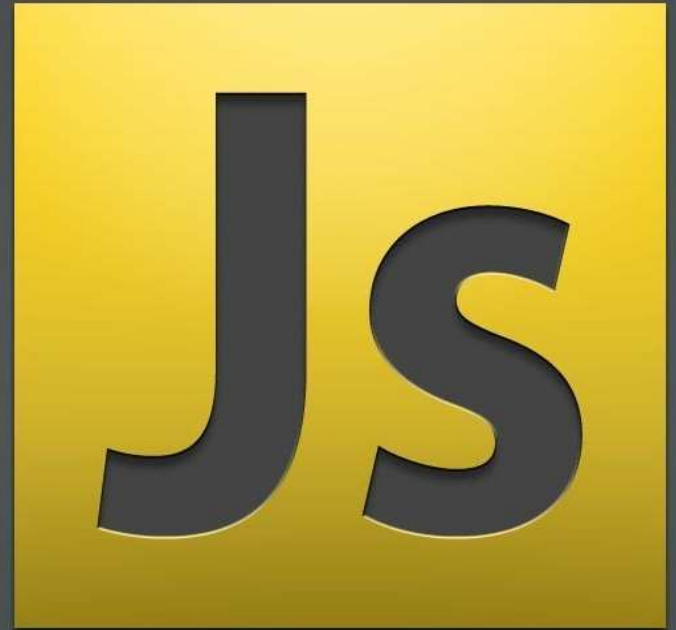
Las herramientas del desarrollador de los navegadores disponen de un apartado para la depuración del código.





# 6

## TRANSPILADORES Y POLYFILLS



Javascript es un lenguaje en continua evolución, de forma que la especificación está continuamente añadiendo nuevas funcionalidades y mejoras.

Por ello, es común que los intérpretes solo implementen una parte del estándar, lo que puede hacer que nuestro código solo funcione en algunos navegadores.

Para solucionar este problema hay dos herramientas:

- Transpiladores
- Polyfills

<https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>

## Transpiladores

Un **transpilador** es un software que traduce código fuente a otro código fuente.

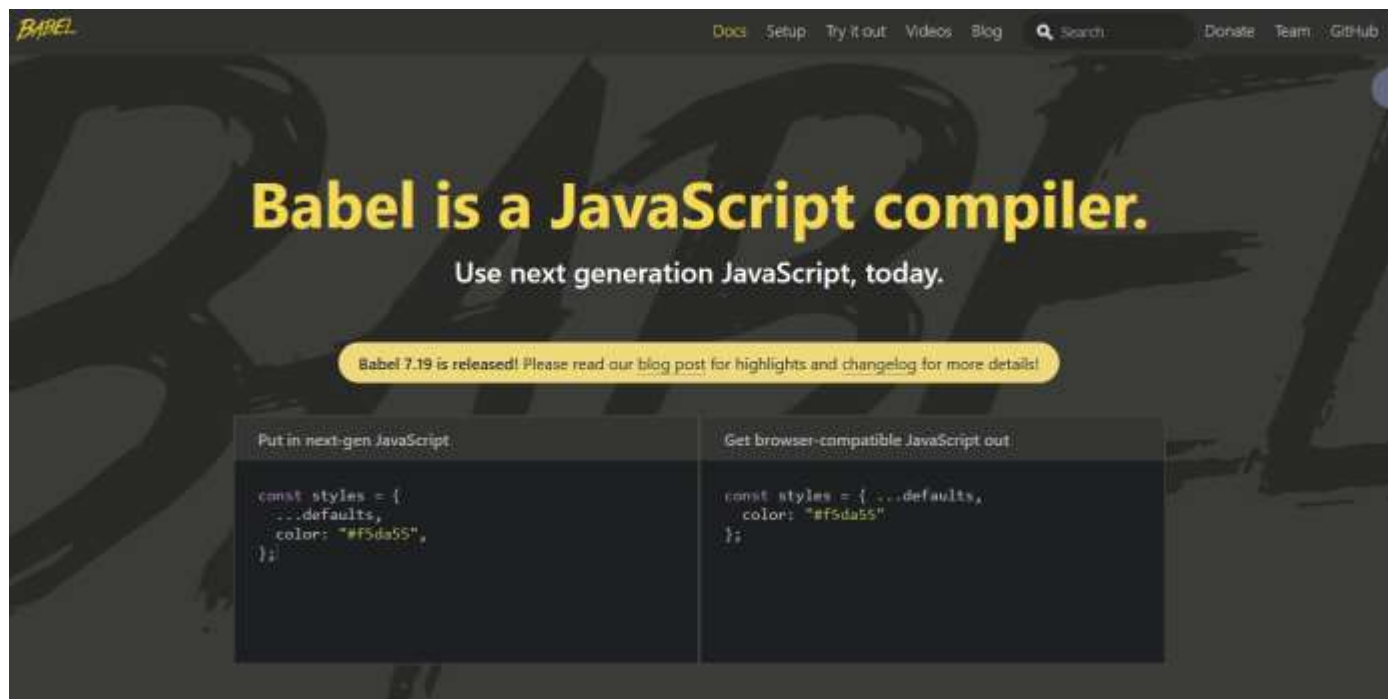
Permite utilizar constructores actuales del lenguaje y generar código que funciona en cualquier navegador aunque sea antiguo y no soporte dichos constructores.

**Ejemplo:** uso de la *coalescencia del nulo*

```
1 // antes de ejecutar el transpilador
2 height = height ?? 100;
3
4 // después de ejecutar el transpilador
5 height = (height !== undefined && height !== null) ? height : 100;
```

Lo habitual es que el desarrollador ejecute el transpilador en su propio equipo y despliegue el código ya transpilado en el servidor.

Uno de los más conocidos es **Babel**.



Generalmente se incluyen en herramientas de compilación (**build systems**) que realizan este proceso automáticamente.



Ejemplos de *builds systems*: Grunt, Broccoli, Webpack o Gulp



## Polyfills

Las actualizaciones del lenguaje no solo incluyen nuevos constructores, sino también **nuevas funciones integradas**.

Un **polyfill** simplemente implementa dichas funciones y las utiliza si el navegador no las soporta.

**Ejemplo:** incluiría este código para la función `Math.trunc`

```
1  if (!Math.trunc) { // no existe tal función
2    // implementarla
3    Math.trunc = function(number) {
4      // Math.ceil y Math.floor existen incluso en los intérpretes antiguos
5      // los cubriremos luego en el tutorial
6      return number < 0 ? Math.ceil(number) : Math.floor(number);
7    };
8  }
```

Ejemplos de estas librerías son **Pollyfill.io** o **core.js**

