**TECNOLÓGICO NACIONAL DE MÉXICO**

INSTITUTO TECNOLÓGICO DE CHIHUAHUA II

**INGENIERÍA EN SISTEMAS COMPUTACIONALES**

Programación web Front end

**Javascript**

**Equipo:** Bubbaloo

Dafne Anaïs Bolaños Morales **#19550743**

Jese David Girón Villegas **#19550735**

Luis Carlos Mata Yañez **#18550717**

**“**Se vienen cosas grandes**”**

**Docente:** Carlos Humberto Rubio Rascon

**Índice**

[**Introducción**](#_hrwowoccqnn8) **3**

[**Historia de Javascript**](#_kjyggswqm73o) **4**

[ECMAScript](#_jxe1gilhhyyf) 5

[**NodeJS**](#_jou4tmlewvyt) 6

[**Conceptos generales**](#_jvg1ehsi28z5) **6**

[Variables](#_2chcaia4ey2k) 6

[Comentarios](#_ibr3hd8xgwdq) 7

[Operadores](#_o6rujpswpfd6) 7

[Condicionales](#_yl2oxaozxaik) 7

[Funciones](#_r7pb5p6arpfr) 8

[Evento](#_ltkgx33bsv6t) 8

[**Bibliotecas JavaScript más utilizadas**](#_vrutu9bc0l33) **8**

[jQuery](#_rznnzwt7o0ev) 8

[jQuery UI](#_bhygzib2pj1j) 9

[Dojo Toolkit](#_kedxd7cqbhen) 9

[React](#_lt26z7asfofr) 9

[Zepto](#_ss8u05tv8gvn) 9

[CreateJS](#_3oj9vs8pqlus) 9

[**Frameworks JS más utilizados**](#_15r0uf4ze5z9) **10**

[AngularJS](#_pco5bgs2xyk1) 10

[Angular](#_q5x72kdxcsxk) 10

[Ember.js](#_52ts331u35ib) 10

[Vue.js](#_gyo94iosxm8m) 10

[Meteor](#_ktmmqre2ddxq) 11

[**Conclusión:**](#_c8fst8wpl290) **11**

**Referencias 11**

# Introducción

JavaScript es conocido como un lenguaje de scripting (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat.

JavaScript es el único lenguaje de programación que funciona en los navegadores de forma nativa (lenguaje interpretado sin necesidad de compilación). Por tanto, se utiliza como complemento de HTML y CSS para crear páginas webs.

A Javascript se le denomina lenguaje "del lado del cliente" porque se ejecuta en contexto del navegador (cliente web), en contraposición a lenguajes como PHP, que se ejecutan del "lado del servidor". En el lado del cliente es el navegador el que soporta la carga de procesamiento y también es el que nos aporta los recursos con los que contamos para programar las aplicaciones.

Desde hace años, gracias a su compatibilidad con todos los navegadores existentes en el mercado, se ha convertido en un estándar para la programación del lado del cliente.

Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, con que cuenta este lenguaje es el propio navegador y todos los elementos que hay dentro de una página (que no es poco).

De manera adicional, gracias a las API Javascript de HTML5, que están disponibles en los navegadores actuales de ordenadores y dispositivos, podemos acceder a todo tipo de recursos adicionales, como la cámara, espacio para almacenamiento de datos, creación de gráficos basados en vectores y mapas de bits, flujos de datos con servidores, etc. Con todo ello se han multiplicado las posibilidades del lenguaje.

Pese a que Javascript es conocido principalmente como un lenguaje del lado del cliente, lo cierto es que Javascript es mucho más. Hoy es posible ampliar el contexto de Javascript a muchos ámbitos. Es especialmente importante porque es el único lenguaje de programación que entienden los navegadores sin necesidad de compiladores, 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.

Es importante no confundir JavaScript con el lenguaje de programación Java. Ambos "Java" y "JavaScript" son marcas o marcas registradas de Oracle en los Estados Unidos y otros países, Sin embargo, los dos lenguajes de programación tienen sintaxis, semántica y usos muy diferentes.

JavaScript se diseñó con una sintaxis similar a C++ y Java, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Pero también JS es el encargado de que existan herramientas como [Google Analytics](https://soyrafaramos.com/que-es-google-analytics-y-como-funciona/), [Google Tag Manager](https://soyrafaramos.com/que-es-google-tag-manager/), Facebook Pixel y tantas otras, que son claros ejemplos de JavaScript.

Existe una tecnología llamada **AJAX** que permite intercambiar información con el servidor sin tener que recargar la página. Esta tecnología desarrollada en JavaScript ha supuesto uno de los principales avances en el desarrollo web y gracias a esta es posible que podamos conseguir mensajes, emails y notificaciones sólo pulsando un botón, sin tener que recargar la página.

# Historia de Javascript

Los primeros años de la web, en los que las páginas eran principalmente contenido y enlaces, puesto que por aquel entonces solo existía HTML. Sin embargo, era necesario aportar algún grado de interacción con el usuario, que fuera capaz de cubrir las necesidades básicas de los desarrolladores en aquella época. Así que se creó un lenguaje capaz de ejecutar pequeños programas en el contexto de una página web, con un juego de instrucciones capaz de interaccionar con el usuario como respuesta a las acciones en la página.

La historia de JavaScript empieza a comienzos de los 90, cuando los usuarios llegan a Internet y acceden a la Web gracias a los navegadores. Las conexiones entre los usuarios y las webs se hacen a través de líneas muy lentas.

Javascript fue creado por Netscape, la compañía propietaria de un navegador con el mismo nombre hoy desaparecido, que fue precursor del actual Firefox. Su lanzamiento se produjo en 1995 en la versión de Netscape 2.0. Su autor, un programador llamado Brendan Eich, se dice que lo creó en una semana.

Originalmente Javascript tomó el nombre de Mocha, aunque antes de su lanzamiento fue renombrado como LiveScript. Sin embargo, ese nombre se cambiaría finalmente a Javascript como consecuencia de un acuerdo entre Netscape y Sun Microsystems, que por aquella época era la propietaria del lenguaje Java.

En virtud de ese acuerdo Netscape agregó compatibilidad para Java en su navegador, a la vez que tomaba el nombre de JavaScript para su lenguaje pues en ese entonces no tenían correlación alguna.

Con el tiempo, las pequeñas diferencias entre JavaScript y JScript se fueron haciendo más patentes y acabó representando un problema para desarrolladores y usuarios, que había que atajar. Para ello en 1997 se produjo un movimiento para la estandarización del lenguaje, que acabó en la creación de ECMAScript, que no es más que el estándar del lenguaje JavaScript.

## ECMAScript

ECMAScript es el estándar creado para homogeneizar lenguaje JavaScript implementado en cada navegador. El trabajo de estandarización de JavaScript comenzó en 1997 y continúa evolucionando hasta la fecha.

La organización que se encarga de esta estandarización es ECMA International, que comenzó el trabajo a raíz de una solicitud de Netscape. El nombre final del lenguaje fue acordado por los distintos fabricantes de navegadores, incluidos Netscape y Microsoft, junto con la propia ECMA sería ECMAScript.

La especificación del estándar más revolucionario ha sido **ECMAScript 6,** también conocida como ES6 y cuyo nombre oficial es ECMAScript 2015, debido a su año de presentación. Esta versión mejoró de manera notable la orientación de objetos de JavaScript, presentando clases y objetos en toda regla. También introdujo los módulos, toda una nueva gama de operadores, funciones lambda, iteradores, etc.

A partir de ES6 se adquirió el compromiso de presentar nuevas especificaciones de ECMAScript cada año.

Las siguientes características son comunes a todas las implementaciones que se ajustan al estándar ECMAScript, a menos que especifique explícitamente en caso contrario:

* **Imperativo y estructurado:** JavaScript hace distinción entre expresiones y sentencias. JavaScript es compatible con gran parte de la estructura de programación de [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)) (por ejemplo, sentencias if, bucles for, sentencias switch, etc.)
* **Dinámicos:**
  + **Tipo Dinámico:** Como en la mayoría de los lenguajes de scripting, el tipo está asociado al valor, no a la variable.
  + **Objetual:** JavaScript está formado casi en su totalidad por objetos. Los objetos en JavaScript son arrays asociativos, mejorados con la inclusión de prototipos. Las propiedades y sus valores pueden ser creados, cambiados o eliminados en tiempo de ejecución.
  + **Evaluación en tiempo de ejecución:** JavaScript incluye la función eval que permite evaluar expresiones expresadas como cadenas en tiempo de ejecución.

#### **Funcional:**

* + **Funciones de primera clase:** A las funciones se les suele llamar ciudadanos de primera clase; son objetos en sí mismos. Como tal, poseen propiedades y métodos, como .call() y .bind().​ Una función anidada es una función definida dentro de otra. Esta es creada cada vez que la función externa es invocada.

#### **Prototípico**

* + **Prototipos:** JavaScript usa prototipos en vez de clases para el uso de herencia.30​ Es posible llegar a emular muchas de las características que proporcionan las clases en lenguajes orientados a objetos tradicionales por medio de prototipos en JavaScript.

### NodeJS

JavaScript ahora mismo es el lenguaje más popular. De hecho, desde hace años se ha creado una versión que es capaz de ser ejecutada también en el lado del servidor (Node JS). Por tanto, ahora mismo se ejecuta JavaScript en los navegadores y en los servidores, creando a su alrededor una amplísima comunidad de desarrolladores casi full-stack.

Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web.

## Conceptos generales

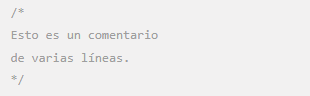
### Variables

Las Variables son contenedores en los que puedes almacenar valores. Primero debes declarar la variable con la palabra clave **var** (menos recomendado) o **let**, seguida del nombre que le quieras dar. Se recomienda más el uso de let que de var.

* **String:** Esto es una secuencia de texto conocida como cadena. Para indicar que la variable es una cadena, debes escribirla entre comillas.
* **Number:** Esto es un número. Los números no tienen comillas.
* **Boolean:** Tienen valor verdadero/falso. true/false son palabras especiales en JS, y no necesitan comillas.
* **Array:** Una estructura que te permite almacenar varios valores en una sola referencia.
* **Object:** Básicamente cualquier cosa. Todo en JavaScript es un objeto y puede ser almacenado en una variable.

### Comentarios

Puedes escribir comentarios entre el código JavaScript. El navegador ignora el texto marcado como comentario. En JavaScript, los comentarios de una sola línea se escriben así:



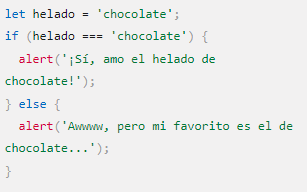
### Operadores

Un operador es básicamente un símbolo matemático que puede actuar sobre dos valores (o variables) y producir un resultado.

* **Suma/concatena + :** Se usa para sumar dos números, o juntar dos cadenas en una.
* **Resta - :** Hace la resta de dos variables.
* **Multiplicación \* :** Hace la multiplicación de dos variables.
* **División / :** Hace la división de dos variables.
* **Asignación = :** Asigna los valores a las variables.
* **Identidad / igualdad === :** Comprueba si dos valores son iguales entre sí, y devuelve un valor de true / false.
* **Negación, distinto ! :** Comprueba que el valor sea diferente.

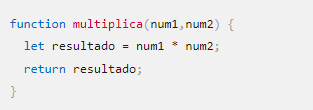
### Condicionales

Las condicionales son estructuras de código que permiten comprobar si una expresión devuelve true o no, y después ejecuta un código diferente dependiendo del resultado. La forma de condicional más común es la llamada if... else. Tambien otro ejemplo es while. ej.



### Funciones

Las funciones son una manera de encapsular una funcionalidad que quieres reutilizar, de manera que puedes llamar esa función con un solo nombre, y no tendrás que escribir el código entero cada vez que la utilices.



### Evento

Para crear una interacción real en tu sitio web, debes usar eventos. Éstos son unas estructuras de código que captan lo que sucede en el navegador, y permite que en respuesta a las acciones que suceden se ejecute un código. El ejemplo más obvio es un clic (click event), que se activa al hacer clic sobre algo.

# Bibliotecas JavaScript más utilizadas

Las librerías JavaScript son un código reutilizable que a menudo tiene un caso de uso principal. Las librerías proporcionan muchas funcionalidades estándar para que los desarrolladores no tengan que preocuparse por muchas funciones. Así, pueden usar estas para crear páginas web fácilmente utilizando componentes de la interfaz de usuario, utilidades de lenguaje, funciones matemáticas y más. Una librería consta de varias funciones, objetos y métodos, según el idioma. Además, las puedes incluir en un proyecto sin depender de una estructura en particular. Es decir, eres libre de usar una, dos o tantas librerías JavaScript como necesites.

Entre las librerías más utilizadas se encuentran:

### **jQuery**

jQuery es la más utilizada de todas las librerías JavaScript. Esto se debe, entre otras razones, a que posibilita la escritura de código jQuery en cualquier tipo de navegador y que para ello existen numerosos plugins. La librería de código abierto está incluida en numerosos sistemas de gestión de contenidos como WordPress, Drupal o Joomla!.

jQuery se utiliza sobre todo como una práctica interfaz de DOM y ofrece numerosas funciones, jQuery resulta especialmente popular debido a la posibilidad de integrar las consultas realizadas con Ajax (solicitudes HTTP que no requieren volver a cargar la página web).

### **jQuery UI**

Entre las funciones de jQuery UI se encuentra la creación de opciones para la interacción (drag and drop, aumento y disminución de elementos web, etc.), de animaciones y efectos, así como de widgets (como Autocomplete, Slider, Datepicker, etc.).

### **Dojo Toolkit**

Está diseñado para realizar aplicaciones web y contenidos web dinámicos. Como una de las bibliotecas de JavaScript más antiguas que todavía sigue teniendo relevancia en el mercado ofrece gran cantidad de funciones adicionales.

### **React**

React fue utilizada por primera vez en 2011 en el newsfeed de Facebook. En 2013 se publicó como una librería JavaScript de código abierto para crear interfaces de usuario. Lo característico de React es que no solo se puede utilizar en clientes de páginas web, sino también en el servidor o durante el desarrollo de aplicaciones.

### **Zepto**

Zepto es una librería JavaScript muy liviana: mientras que jQuery convence, sobre todo, por su funcionalidad, esto mismo es lo que intimida a muchos desarrolladores, y es que dicha biblioteca de JavaScript puede resultar muy compleja. La minimalista Zepto es mucho más ligera, por lo que se carga más rápido y ocupa menos espacio, pudiendo integrarse así en aplicaciones móviles. Para que el volumen de archivos sea mínimo, Zepto reduce, por ejemplo, la compatibilidad con los navegadores más antiguos. Asimismo, para permitir el uso de Ajax y de animaciones, se debe recurrir a módulos adicionales.

### **CreateJS**

No una única biblioteca, sino varias: CreateJS es una suite compuesta por cuatro bibliotecas diferentes: EaselJS (gráficos e interactividad), TweenJS (animaciones), SoundJS (audio) y PreloadJS (preloading). No existe dependencia entre ellas, lo que significa que no es necesario implementar las cuatro en un mismo proyecto cuando solo se necesita una. Además, se ofrecen algunas herramientas para simplificar el trabajo con las librerías de JavaScript. El foco de atención es el desarrollo de aplicaciones HTML5 y Flash. En general, el equipo de CreateJS trabaja estrechamente con Adobe, por lo que también hay herramientas que garantizan la colaboración fluida de la suite con los productos de Adobe.

# Frameworks de JavaScript más utilizados

El número de librerías JavaScript es notablemente mayor que el de frameworks. Estos últimos sirven, sobre todo, para aplicaciones web complejas y son muy efectivos en caso de que los desarrolladores intervengan en la conceptualización y especificación del framework correspondiente.

### **AngularJS**

El framework creado por Google tiene, con diferencia, la mayor comunidad entre los frameworks JavaScript. AngularJS se utiliza para crear aplicaciones web de una sola página (aplicaciones web que constan de un solo documento HTML), a causa del patrón MVVM (Model-View-ViewModel), se pueden desarrollar aplicaciones web especializadas en la interacción con los usuarios.

### **Angular**

Angular, a menudo también llamado Angular 2, es el sucesor de AngularJS. Este framework de JavaScript está orientado principalmente al desarrollo de aplicaciones web de una sola página, aunque Google ha realizado cambios muy importantes en la segunda versión. La mayor diferencia es, en este sentido, que ya no se utiliza JavaScript para programar, sino TypeScript. Dado que el lenguaje de scripts de Microsoft se basa en JavaScript y es compatible con él, no se dan restricciones directas en el desarrollo de JS.

### **Ember.js**

Ember.js también es un framework del lado del cliente con el que se pueden crear aplicaciones web de una sola página y aplicaciones de escritorio. Uno de sus rasgos distintivos con respecto a AngularJS es que los creadores de Ember.js han incluido a la comunidad de una manera más intensiva en el proceso de desarrollo del framework y que los cambios importantes realizados en el framework se discuten en dicha comunidad antes de ser ejecutados. Ember.js se comercializa como un framework con el que se pueden desarrollar aplicaciones web ambiciosas, por lo que está orientado, en primer lugar, a desarrolladores que ya tienen cierta experiencia en la creación de aplicaciones web.

### **Vue.js**

Vue.js es, asimismo, un framework de JavaScript para el desarrollo de aplicaciones web de una sola página que recuerda a Angular y React. Los desarrolladores de este proyecto relativamente reciente han diseñado Vue.js de tal manera que iniciarse con él sea relativamente sencillo. Así es posible, por ejemplo, integrar plantillas en HTML. Asimismo, Vue.js es mucho más flexible que otros frameworks que normalmente prescriben de una forma un tanto rígida cómo utilizar el framework.

### **Meteor**

Meteor, también conocido como MeteorRJ, es un framework JS especialmente apto para el desarrollo cross-platform que permite a los desarrolladores crear aplicaciones web y móviles con el mismo código. Otra ventaja es que los cambios en el código pueden enviarse directamente a los clientes gracias a un Distributed Data Protocol (DDP) propio.

# Conclusión:

Aunque JavaScript surgió con el objetivo inicial de programar ciertos comportamientos sobre las páginas web, respondiendo a la interacción del usuario y la realización de automatismos sencillos, lo cierto es que hoy sus ámbitos de trabajo son mucho mayores. Ha dejado de ser un "lenguaje de scripting" del lado del cliente, para convertirse en un lenguaje integrador de múltiples ámbitos y prestaciones.

Encontramos Javascript, ya no solo en la Web, también es nativo en sistemas operativos como Windows. También es capaz de hacer programas de consola, bajo la plataforma NodeJS, así como programas de escritorio multiplataforma (Windows, Linux y Mac). Paralelamente, se puede usar Javascript para el desarrollo de aplicaciones para dispositivos (apps híbridas) y aplicaciones que compilan a nativo.

# Referencias

MDN contributors (4 de agosto de 2022) “JavaScript” de developer.mozilla.org. Recuprado el 31 de agosto de 2022 de: <https://developer.mozilla.org/es/docs/Web/JavaScript>

Wikipedia (27 de agosto de 2022) “JavaScript” de wikipedia.org. Recuperado el 31 de agosto de 2022 de: <https://es.wikipedia.org/wiki/JavaScript>

Desarrolloweb (S.F) “Javascript” de desarrolloweb.com. Recuperado el 31 de agosto de 2022 de: <https://desarrolloweb.com/home/javascript>

Rafa Ramos (S.F) “¿Qué es JavaScript y para qué sirve?” de soyrafaramos.com. Recuperado el 31 de agosto de 2022 de: <https://soyrafaramos.com/que-es-javascript-para-que-sirve/>

MDN contributors (18 de agosto de 2022) ”Fundamentos de javascript” de developer.mozilla.org. Recuperado el 31 de agosto de 2022 de: <https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/JavaScript_basics>

Miriam Martinez Canelo (8 de julio de 2020) “12 librerías JavaScript que deberías conocer” de profiles.es. Recuperado el 31 de agosto de 2022 de: <https://profile.es/blog/librerias-javascript/#Momentjs>

Desarrollo web (10 de mayo de 2019) “Las librerías y los frameworks JavaScript más populares” de ionos.mx. Recuperado el 31 de agosto de 2022 de: <https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/frameworks-javascript-y-librerias-populares/>

Recomendación: <https://developer.mozilla.org/es/docs/Web/JavaScript/Guide>