ÍNDICE

INTRODUCCIÓN

INTRODUCCIÓN A JAVASCRIPT

Esta obra está dirigida a todas aquellas personas que quieran aprender a programar con JavaScript. El libro es ante todo un texto práctico que explica mediante numerosos ejemplos los principales elementos del lenguaje.

Se recomienda tener conocimientos básicos de HTML y del entorno de programación web para poder seguir sin problemas este curso. No obstante, la orientación de los diferentes capítulos permite que lector vaya adquiriendo paulatinamente los conocimientos necesarios para ir comprendiendo el lenguaje de programación JavaScript partiendo prácticamente desde cero.

Cómo trabajar con este libro

Para aprovechar al máximo este curso, se recomienda seguir todos los capítulos por orden y realizar los distintos ejercicios que van apareciendo. Probar los ejemplos, modificarlos y ver cómo actúa el programa es una forma estupenda de empezar a familiarizarse con JavaScript.

Se recomienda usar navegadores de última generación para la visualización de los ejercicios y programas que van apareciendo a lo largo del curso.

Una de las grandes ventajas de JavaScript es que no necesita ningún software especial para programar. Basta un simple editor de texto para empezar a trabajar.

Sin embargo, el hecho de que podamos usar ese editor de texto para escribir código, no quiere decir que sea la única manera.

Para trabajar óptimamente se recomienda el uso de un entorno de desarrollo integrado llamado IDE (*Integrated Development Environment*), que no es más que una aplicación que proporciona los medios para facilitar al programador su trabajo.

Visual Studio Code (VS Code) en estos momentos es posiblemente el mejor IDE de JavaScript para Windows, Mac y Linux y aparte de ser compatible con JavaScript, también lo es con Node.js y TypeScript, además de disponer de extensiones para otros lenguajes como C ++, C #, Python, PHP, etc.

VS Code facilita enormemente la tarea de programación. Es una utilidad que dispone de la función de autocompletado, colorea el código de los scripts, permite trabajar con varios documentos, ofrece ayuda, etc. En resumen, una herramienta que facilita la tarea de programación de los scripts. El **apéndice B** de este libro está dedicado a esta aplicación con la que el lector podrá adquirir los conocimientos básicos para empezar a utilizarla.

Hay disponibles también editores de texto avanzados como Atom, Notepad++ o Sublime Text, que cuentan con características similares y que son de gran ayuda para el desarrollo de código.

Por otro lado, todos los navegadores actuales disponen de una herramienta de depuración llamada **consola**. La consola de JavaScript será útil para chequear los errores de código, leer variables en tiempo de ejecución o, incluso, inspeccionar elementos de la página.

Mientras que la elección del editor de código la dejamos en manos del lector en función de sus preferencias, en el capítulo siguiente entraremos en detalle con el uso de la consola de JavaScript ya que es una herramienta de gran utilidad.

Convenciones utilizadas en este libro

En este libro se utilizan una serie de convencionalismos que permiten hacer más claros la lectura de la obra y el seguimiento del curso.

Veamos a continuación cuáles son:

Cuando aparezca este icono en algún punto del texto, es importante prestar atención. Encontraremos una advertencia que evitará futuros problemas o quebraderos de cabeza.

Este icono indica que nos encontramos antes una nota curiosa, idea o comentario que puede resultar útil al lector. Se trata de información importante para tener en cuenta.

Los códigos de ejemplo que van a ir saliendo a lo largo del curso aparecerán de esta forma:

SCRIPT X.X. EJEMPLO DE PROGRAMA CON JAVASCRIPT

```
<!DOCTYPE html>
<html>
  <head>
     <meta charset="UTF-8">
      <title>Ejemplo de código</title>
       <script>
       alert ("Hola Mundo");
       </script>
    </head>
  <body>
  </body>
</html>
```

El primer número indicará el capítulo donde se desarrolla el código, y el segundo, el número de orden en relación con dicho capítulo.

Material adicional

Todos los ejemplos prácticos de este libro están disponibles para su descarga en la web de la **Editorial RC Libros** (http://www.rclibros.es).

Para descargarlos, el lector deberá acceder a la ficha de este libro en la web de RC Libros donde encontrará las instrucciones de descarga.

Agradecimientos

Escribir un libro implica consumir parte del escaso y valioso tiempo libre disponible para la familia; por este motivo, deseo reconocer una vez más la comprensión y el apoyo incondicional de mi esposa Nieves y de mis hijas Lydia y Sonia por haber aguantado una vez más mi encierro en el estudio mientras escribía este libro.

Gracias a todos mis lectores de España e Hispanoamérica por sus felicitaciones y críticas constructivas. Hace más de 10 años de mi última publicación y los ánimos recibidos durante todos estos años contribuyen sin duda a pensar en nuevos proyectos.

Y finalmente, dar las gracias a todo el equipo de **RC Libros** y en especial a José Luis Ramírez, por seguir confiando en mi trabajo después de tantos años.

Gracias a todos.

Juan Carlos Orós

1 JAVASCRIPT

QUÉ ES JAVASCRIPT

JavaScript es un lenguaje de programación que funciona en los navegadores de forma nativa. Es un lenguaje interpretado que no necesita ser compilado y se utiliza como complemento de HTML y CSS para crear páginas web.

No hay que confundir Java con JavaScript. Java es un lenguaje completo que permite crear aplicaciones independientes y necesita ser compilado, mientras que JavaScript es un lenguaje que funciona como extensión del HTML. Es un lenguaje de programación orientado a objetos, diseñado para el desarrollo de aplicaciones cliente-servidor a través de Internet.

Este lenguaje fue desarrollado originalmente por Brendan Eich y propusieron JavaScript para que fuera adoptado como estándar de la ECMA **(European Computer Manufacturers Association).** En junio de 1997 fue adoptado como un estándar ECMA, con el nombre de ECMAScript.

ECMAScript es la especificación donde se mencionan todos los detalles de cómo debe funcionar y comportarse JavaScript en un navegador. De esta forma, los diferentes navegadores (Chrome, Safari, Firefox, etc.) saben cómo se deben desarrollar los motores de JavaScript para que cualquier código o programa funcione de acuerdo con el estándar independientemente del navegador que se esté utilizando.

En el momento de la redacción de este manual, la versión actual del estándar es la JavaScript **ES12 o ECMAScript 2021**, desarrollada en GitHub con la ayuda de la comunidad ECMA International.

El campo de actuación de JavaScript es muy amplio y proporciona alternativas para muchos **entornos** de ejecución, pero sin duda, uno de los más extendidos es el de la programación para entornos web, objetivo de este libro.

JavaScript es un lenguaje de scripting del lado del cliente, por lo tanto interpretación y ejecución es responsabilidad del navegador (cliente) y no del servidor como sucede por ejemplo con PHP.

El código de programa de JavaScript, llamado script, se introduce directamente en el documento HTML y no necesita ser compilado, es el propio navegador el que se encarga de traducir dicho código. Por este motivo, el código fuente del programa debe descargarse del navegador junto con el resto del documento web para poder ser interpretado.

Este motivo que puede parecer una desventaja es en realidad una ventaja ya que el procesar complejos programas en el navegador no sobrecarga el trabajo de los servidores pues podemos desarrollar programas que se ejecuten directamente en el navegador (cliente) de manera que este pueda efectuar determinadas operaciones o tomar decisiones sin necesidad de acceder al servidor.

En resumen, con el uso de JavaScript es posible dinamizar una página HTML, interactuar con el usuario y automatizar procesos en función de eventos relacionados con el comportamiento del usuario y su interacción con el contenido del documento web.

Serviría como ejemplo algo tan simple como mostrar una fecha en el documento o que en la página se verifique una clave de acceso para poder acceder a una determinada web.

Dado que no todos los navegadores soportan JavaScript de la misma forma lo mejor es comprobar qué versión implementa cada uno de ellos. Para ello, todos los navegadores disponen de un comando en línea que permite ver entre otros, el estándar de JavaScript que soportan.

En el caso de Chrome, poniendo en la barra de dirección ***chrome://versión***, accederemos a toda la información relativa a los diferentes motores que usa el navegador.

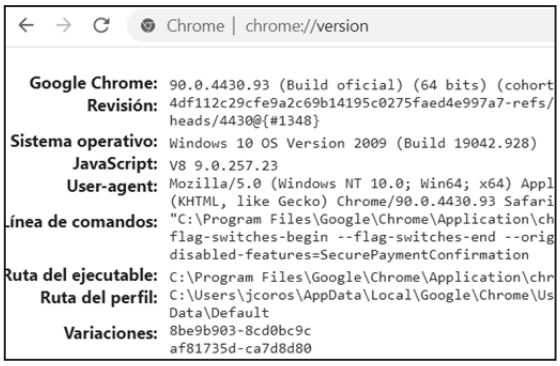

Figura 1.1. Navegador Chrome mostrando la versión de JavaScript que soporta. En este caso la V8 9.0.257.23

PRIMEROS PASOS

Vamos a comenzar con el estudio de JavaScript. Para ello necesitamos conocer la nada difícil sintaxis del lenguaje.

Sabemos que un programa en JavaScript no debe ser compilado, ya que el navegador se encarga de leerlo cuando se carga la página, pero esto no quiere decir que todos los códigos JavaScript se ejecuten nada más cargarse el documento HTML, algunas funciones permanecen en espera hasta que el usuario realiza una acción determinada, como podría ser pulsar sobre un enlace.

La ubicación más normal para ubicar el script en un documento es dentro de la sección <head> con el uso de las etiquetas <script> y </script>, pero realmente, el código JavaScript se puede insertar en la cabecera o entre las etiquetas del cuerpo del documento <body>.

La elección de dónde implementar el código dependerá de los eventos o acciones que debe tomar la página en función de la interacción del usuario, pero como norma general podríamos decir que los scripts que producen un evento, habitualmente, se incluyen en el cuerpo de la página mientras que las bibliotecas o funciones generales se insertan en el encabezado del documento.

JavaScript es **case sensitive**, es decir, es capaz de diferenciar las mayúsculas de las minúsculas, por lo tanto, como veremos más adelante, variables llamadas **DATO, Dato** o **dato**, son diferentes a los ojos de JavaScript.

La etiqueta <script>

La etiqueta <script> se utiliza para insertar o hacer referencia a un script ejecutable dentro de un documento HTML o XHTML. En HTML 4 es necesario especificar mediante el atributo *type* el lenguaje en que está escrito el código embebido dentro de la etiqueta <script>.

```
<script type="text/javascript">
```

Sin embargo, en HTML 5 es opcional, por lo tanto, en el resto del libro será implementada de acuerdo con el estándar de HTML 5 al considerarse la opción anterior obsoleta.

Veamos a continuación un ejemplo básico de la estructura de un documento HTML con nuestro primer código JavaScript.

SCRIPT 1.1. MI PRIMER PROGRAMA CON JAVASCRIPT

```
<!DOCTYPE html>
    <html>
    <head>
        <meta charset="UTF-8">
        <title>1.1 - Ejemplo de código JavaScript</title>
        <script>alert ("Hola Mundo");</script>
    </head>
    <body></body>
    </html>
```

En el ejemplo vemos que el código JavaScript se ha colocado en la cabecera del documento mediante la etiqueta <script> y aunque se tratará más adelante, vemos que se ha utilizado la función *alert* que, al ejecutarse, abre una ventana en el navegador mostrando el superconocido “Hola Mundo”.

Figura 1.2. Navegador mostrando el resultado de ejecutar la función alert

La sintaxis anteriormente mostrada es la forma más sencilla de añadir JavaScript en el propio código fuente del documento HTML, pero también podemos agregar código desde un archivo externo que tenga la extensión **.js.** Esta opción permite separar el código de programación JavaScript del código HMTL del documento, funcionalidad muy útil cuando se trabaja con scripts muy pesados o repetitivos.

Para incluir archivos JavaScript externos utilizaremos el atributo **src**, al que se le indicará la URL del archivo **js.**

```
<script src="holamundo.js"></script>
```

También podemos indicar en nuestro documento, el punto exacto donde queremos que aparezca un determinado elemento, como podría ser el texto "Hola Mundo" anteriormente comentado. El código deberá implementarse en el <body> de la página en el punto exacto donde desea que aparezca la cadena.

```
<body><script>
document.write("Hola Mundo");
</script></body>
```

Observe que se ha insertado un punto y coma al final de la instrucción, algo que no es obligatorio la mayoría de las veces ya que el intérprete de JavaScript reconoce el final de una línea de código, pero es recomendable insertarlo por seguir buenas prácticas de programación y obligatorio si se escriben varias instrucciones en una misma línea.

Veamos un ejemplo:

```
<body><script>
document.write("Hola"); document.write(" Mundo");
</script></body>
```

La etiqueta <noscript>

Para evitar errores en navegadores obsoletos o si el usuario ha desactivado voluntariamente el soporte JavaScript, es posible implementar la etiqueta <noscript>. Su misión es definir un contenido alternativo en el documento web cuando el navegador no soporta la ejecución de scripts o bien, cuando el usuario los ha desactivado en el navegador.

La etiqueta no tiene atributos y está soportada prácticamente la mayoría de los navegadores. Por ejemplo, podemos mostrar un texto diciendo “Su navegador no soporta JavaScript” o cualquier otra secuencia de comandos alternativos.

```
<noscript>
Su navegador no soporta JavaScript.
</noscript>
```

Comentarios

Como todo lenguaje de programación, en códigos extensos o incluso en sencillos, se hace necesaria la inclusión de comentarios aclaratorios para futuras revisiones.

Agregar comentarios es una buena práctica de programación que todo buen desarrollador debería seguir. Su uso debe considerarse indispensable cuando se trabaja en grupo, pero abusar de ellos tampoco es recomendable. Busque el equilibrio.

Para introducir comentarios en JavaScript podemos utilizar dos técnicas diferentes:

- Los comentarios en una sola línea irán precedidos de //.
- Los comentarios de varias líneas irán encerrados entre /* y */.

Veamos como ejemplo un fragmento de código en el que se utilizan los sistemas de comentarios anteriormente explicados:

```
<script>
//Comentario de una sola línea
//Podemos incluir también código (que no se ejecutará)
//alert ("Hola Mundo")
//
/* Esta es otra forma de introducir comentarios
pero utilizando varias líneas.
*/
</script>
```

CONSOLA DEL NAVEGADOR

A medida que vayamos avanzando con los conocimientos de JavaScript, los programas se van a ir complicando y tendrán más líneas de código, por lo que va a resultar imprescindible disponer de una herramienta que permita depurar el código, ver errores, etc.

La consola del navegador es una herramienta perfecta que nos va a ayudar a depurar nuestros scripts. Por suerte, todos los navegadores modernos incorporan una consola cómo herramienta de ayuda a los desarrolladores.

La consola de JavaScript nos será útil para chequear los errores de nuestro código, leer variables en tiempo de ejecución para comprobar su valor o inspeccionar elementos de la página.

Todos los navegadores modernos incorporan una consola como herramienta de ayuda a los desarrolladores. Podemos activarla desde el menú del navegador o con una combinación de teclas. Para los navegadores más usados, las combinaciones de teclas son:

- **Google Chrome**
  - Microsoft: Ctrl + Shift + J
    - También Ctrl + Shift + I
    - F12
  - MacOS: Cmd + Option + J

- **Microsoft Edge**
  - Presiona F12
- **Mozilla Firefox**
  - Microsoft: Ctrl + Shift + K
  - MacOS: Cmd + Option + K
- **Safari**
  - Cmd + Option + C

El funcionamiento de la consola es similar en todos los navegadores, aunque existen pequeñas diferencias en el entorno de gestión. Hay que destacar que todos los comandos y métodos que vamos a ver ahora, sí funcionarán de la misma forma en diferentes navegadores ya que son propios del lenguaje JavaScript que todos ellos soportan.