JAVASCRIPT

Ahora te encuentras en el inicio de la documentación que trata sobre el lenguaje Web Javascript. Durante la lectura aprenderás a hacer dinámicas tus páginas web y hacerlas mucho más atractivas para los visitantes.

Esta documentación cubrirá muchos temas, desde lo básico. Aprenderás cómo hacer animaciones, aplicaciones complejas y utilizar este lenguaje junto con HTML5, la nueva versión del famoso W3C.

Esta documentación discutirá principalmente el uso de JavaScript en un entorno de navegador Web, por lo que es esencial que sepas codificación HTML y CSS. Conocer PHP puede ser una ventaja.

Parte 1: Conceptos básicos de Javascript

Como cualquier otro lenguaje de programación, JavaScript tiene algunas características especiales: sintaxis, modelo de objetos, etc. Claramente, cualquier cosa que diferencia un lenguaje de otro. Además, descubrirás rápidamente que JavaScript es un lenguaje relativamente especial en su acercamiento a las cosas. Esta parte es esencial para cualquier principiante de programación e incluso para aquellos que ya conocen un lenguaje de programación debido a que las diferencias con otros lenguajes de programación son numerosas.

Introducción a JavaScript

Antes de entrar directamente en el núcleo de la cuestión, este capítulo te enseñará lo que Javascript, puede hacer, cuando se puede o se debe utilizar y cómo ha evolucionado desde su creación en 1995.

También vamos a discutir algunos conceptos básicos tales como las definiciones exactas de ciertos términos.

¿Qué es JavaScript?

JavaScript es un lenguaje de programación de *scripts* (secuencia de comandos) orientado a objetos. Esta descripción es un poco rudimentaria, hay varios elementos que vamos a diseccionar.

- Un lenguaje de programación

En primer lugar, un **lenguaje de programación** es un lenguaje que permite a los desarrolladores escribir código fuente que será analizado por un ordenador.

Un **desarrollador** o programador es una persona que desarrolla programas. Puede ser un profesional (un ingeniero, programador informático o analista) o un aficionado.

El **código fuente** está escrito por el desarrollador. Este es un conjunto de acciones, llamadas instrucciones, lo que permitirá dar órdenes al ordenador para operar el programa. El código fuente es algo oculto, como un motor en un automóvil está oculto, pero está ahí, y es quien asegura que el coche puede ser conducido. En el caso de un programa, es lo mismo, el código fuente rige el funcionamiento del programa.

Dependiendo del código fuente, el ordenador realiza varias acciones, como abrir un menú, iniciar una aplicación, efectuar búsquedas, en fin, todo lo que el equipo es capaz de hacer.

Hay una gran cantidad de lenguajes de programación, la mayoría se encuentran en esta página de la Wikipedia.

- *Scripts* de programación

JavaScript te permite programar *scripts*. Como se mencionó anteriormente, un lenguaje de programación es utilizado para escribir código fuente a ser analizada por un ordenador. Hay tres formas de usar el código fuente:

Lenguaje compilado como: El código fuente se da a un programa llamado compilador que lee el código fuente y lo convierte en un lenguaje que el equipo será capaz de interpretar: el lenguaje binario, es de 0 y 1. Lenguajes como C o C ++ son lenguajes compilados muy conocidos.

Lenguaje precompilado: aquí, el código fuente se compila en parte, por lo general en un código más fácil de leer para el ordenador, pero que todavía no es binario. Este código intermedio es para ser leído por lo que se llama una “Máquina Virtual", que ejecutará el código. Lenguajes como C # o Java se llaman precompilados.

Lenguaje interpretado: en este caso, no hay compilación. El código fuente se mantiene sin cambios, y si desea ejecutar este código, debemos proporcionar un intérprete que va a leer y realizar las acciones solicitadas.

Los scripts son en su mayoría interpretados. Y cuando decimos que JavaScript es un lenguaje interpretado, lo que significa que es un lenguaje interpretado. Por tanto, es necesario contar con un intérprete para ejecutar código Javascript, y el intérprete que se utiliza una frecuencia: se incluye en tu navegador de internet.

Cada navegador tiene un intérprete Javascript, que varía en función del navegador. Si está utilizando Internet Explorer, el intérprete es llamado JScript (versión 9 intérprete llamado Chakra), en Mozilla Firefox se llama SpiderMonkey y el motor V8 es el de Google Chrome.

- Lenguaje orientado a objetos

Queda una specto a analizar: **orientado a objetos**. Este concepto es bastante complicado de configurar ahora y se profundizará sobre todo después de la parte 2. Sin embargo, un lenguaje de programación orientado a objetos es un lenguaje que contiene elementos, llamados objetos y los objetos diferentes tienen características específicas y formas de uso diferente. El lenguaje proporciona objetos básicos, como imágenes, fechas, cadenas de caracteres... También es posible crear tus propios objetos para hacer la vida más fácil y obtener un código fuente más claro (fácil de leer) y una forma de programar mucho más intuitivo (lógica).

Es muy probable que no entiendas este paso si nunca ha realizado programación, pero no: comprenderás muy pronto cómo funciona todo.

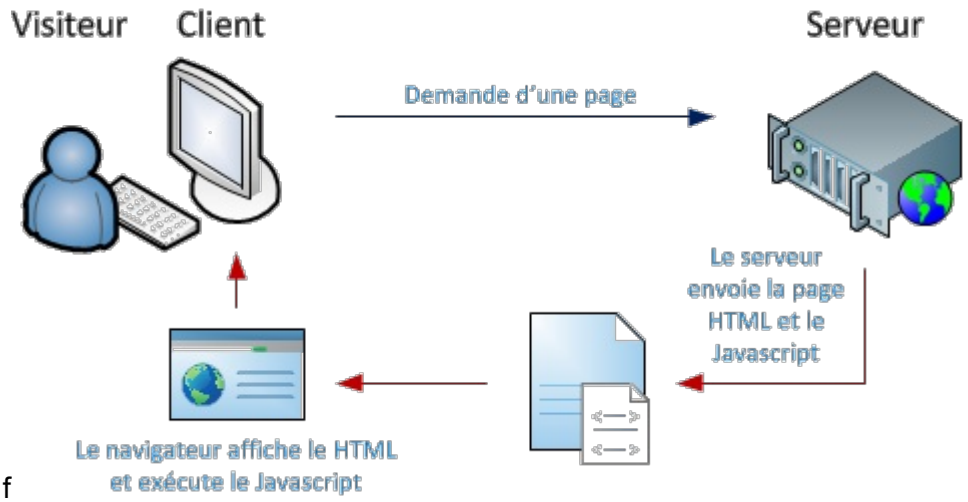
JavaScript, el lenguaje de scripts

Javascript actualmente es principalmente utilizado en internet, junto con las páginas web (HTML o XHTML). Javascript está directamente incluido en la página web (o en un archivo externo) y mejora una página HTML, añadiendo interacción del usuario, animación, ayudas a la navegación, tales como:

Mostrar / ocultar el texto;
Deslizamiento de imágenes;
Crear presentaciones de diapositivas;
Crear burbujas de información.

De JavaScript se dice que es un lenguaje del lado del cliente, es decir que los scripts son ejecutados por el navegador del usuario (cliente). Esto difiere de los llamados lenguajes de script del lado del servidor que son ejecutadas por el servidor web. Este es el caso de lenguajes como PHP.

Esto es importante porque el propósito de los scripts del lado del cliente y del lado del servidor no es el mismo. Un script del lado del servidor se encargará de "crear" la página web que se envía al navegador. Este entonces mostrará la página a continuación, ejecutará secuencias de comandos del lado del cliente como JavaScript. Un patrón que se repite en esta operación:



f

Javascript no es la Web

Si Javascript está diseñado para ser usado en conjunción con HTML, el lenguaje ha evolucionado desde entonces hacia otros destinos. Javascript es regularmente utilizado para hacer extensiones para diferentes programas, como los scripts codificados en Lua o Python.

JavaScript también se puede utilizar para construir aplicaciones. Mozilla Firefox es el ejemplo más famoso: la interfaz del navegador se crea con una especie de HTML llamado XUL y JavaScript que se utiliza para animar la interfaz. Otros programas también están basados en esta tecnología, como por ejemplo de *TomTom HOME* que se utiliza para administrar tu navegador GPS TomTom a través de tu PC.

Breve historia del lenguaje

En 1995, Brendan Eich trabajaba en *Netscape Communications Corporation*, la compañía que publicó el famoso Netscape Navigator, entonces principal competidor de Internet Explorer. Brendan desarrolló Live Script un lenguaje de script que se basa en el lenguaje Java, y que estaba destinado a ser instalado en los servidores desarrollados por Netscape. Netscape inició el desarrollo de una versión del cliente LiveScript, que pasó a llamarse JavaScript, en homenaje al lenguaje Java creado por Sun Microsystems.

En efecto, en ese momento, el lenguaje Java era cada vez más popular, y Brendan Eich, el padre de Javascript, al llamarlo JavaScript en lugar de LiveScript era una forma de publicidad de Java y JavaScript en sí. Atención, al final, estos dos lenguajes son radicalmente diferentes, no vayas a confundir Java y JavaScript, pues operan de forma diferente.

Javascript fue lanzado en diciembre de 1995 y estaba integrado en Netscape Navigator 2. El lenguaje fue tan exitoso, por lo que Microsoft desarrolló una versión similar llamada JScript, que se instalaba en Internet Explorer 3, en 1996. Netscape decidió enviar a su versión de Javascript a ECMA International (*European Computer Manufacturers Association*, la Asociación Europea

de Normalización de hoy los sistemas de información y comunicación) para que el lenguaje fuera normalizado, es decir para que se creara una referencia del lenguaje y que así puediera ser utilizado por otras personas y embebidos en otro *software*. ECMA International estandarizó el lenguaje con el nombre de ECMAScript. Desde entonces, las versiones de ECMAScript han evolucionado. La versión más conocida es utilizada en todo el mundo, es la versión ECMAScript 3, publicado en diciembre de 1999.

ECMAScript y sus derivados

ECMAScript es la línea de base en el flujo de implementaciones de referencia. Obviamente, se puede citar a Javascript, que se implementa en la mayoría de los navegadores, pero también:

JScript, que está embebido en la aplicación Internet Explorer. También es el nombre del intérprete de Internet Explorer;
JScript.NET, que se inserta en el marco de Microsoft NET.;
ActionScript, que es la implementación realizada por Adobe en Flash;
EX4 que es el desarrollo de gestión de ECMAcript de XML en el seno del intérprete JavaScript
SpiderMonkey,de Firefox.

Versiones de Javascript

Las versiones de Javascript se basan en los de la ECMAScript (que abreviaremos como ES). Por lo tanto, se encuentran:

ES 1 y ES 1, que son los inicios de Javascript;
ES 3 (publicada en diciembre de 1999), que es funcional en todos los navegadores (excepto las versiones anteriores de Internet Explorer);
ES 4, que fue abandonada debido a los grandes cambios que no fueron apreciados;
ES 5 (publicada en diciembre de 2009), que es la versión más reciente liberada;
ES 6, que se encuentra actualmente en fase de diseño.

Esta documentación cubrirá todas las versiones actualizadas.

Un logotipo desconocido

No hay imágenes oficiales para representar Javascript. Sin embargo, este logotipo se utiliza cada vez más por la comunidad, especialmente desde su introducción en EE.UU. en JSConf EU. Se puede encontrar en esta dirección en diferentes formatos.

Resumen

- JavaScript es un lenguaje de programación interpretado, es decir, que necesita un intérprete para ser ejecutado.

- JavaScript se utiliza principalmente en páginas web.
- Al igual que HTML, JavaScript es ejecutado por el navegador del usuario: se llama un de cliente, en comparación con el lado del servidor cuando el código es ejecutado por el servidor.
- Javascript está normalizado por ECMA International como el nombre *ECMAScript Language Reference*.
- Hay otros lenguajes derivadas del ECMAScript como ActionScript, EX4 o JScript.NET.
- La última versión del estándar está basado en ECMAScript 5, lanzado en 2009.

Primeros pasos en Javascript

Como se mencionó anteriormente, JavaScript es un lenguaje utilizado principalmente con el lenguaje HTML, en este capítulo se aprende cómo integrar este lenguaje en tus páginas web, descubrir su sintaxis básica y mostrar un mensaje en la pantalla del usuario.

También se encuentran al final de este capítulo algunos enlaces que pueden probablemente ser útiles durante la reproducción de esta documentación.

En cuanto al editor de texto a usar (en el que se escribe el código Javascript) es muy probable que valga el que se ha empleado para el código HTML.

Muestra un cuadro de diálogo

Hola Mundo! No se deroga la regla tradicional de que todos los tutoriales de programación comenzarán mostrando el texto "*Hello World!*", ("¡Hola Mundo!" en español) al usuario. A continuación se muestra un programa HTML simple que contiene la instrucción Javascript, situada dentro de un elemento <script>:

Código: HTML - Hola Mundo!

```
<!DOCTYPE html>
<html>
<head>
<title>Hello World!</title>
</head>
<body>
        <script>
        alert('Hello world!');
        </script>
</body>
</html>
```

Apareciendo al ejecutarlo la siguiente caja de diálogo:

Lo nuevo

En el código HTML indicado anteriormente, vemos algunas nuevas características. En primer lugar, un elemento <script> está presente: es él quien tiene el código javascript de la siguiente manera:

Código: JavaScript

```
alert ('Hello world');
```

Es una declaración, es decir, una orden, o más bien una acción que el equipo tendrá que hacer. los lenguajes de programación consisten en una secuencia de instrucciones que, colocados de extremo a extremo, permiten obtener un programa o un script completo. En este ejemplo, no es una instrucción: se llama a la función alerta.

El cuadro de diálogo de alert ()

alert () es una declaración simple, llamada función, que muestra un cuadro de diálogo que contiene un mensaje. este mensaje se coloca entre comillas, entre los paréntesis de la function alert ().

Sintaxis de Javascript

Instrucción

La sintaxis de Javascript no es complicada. Generalmente, las instrucciones deben estar separadas por un punto y coma que se coloca al final de cada instrucción:

Código: JavaScript

```
sentencia_1;
sentencia_2;
sentencia_3;
```

En realidad, el punto y coma no es necesario si la instrucción siguiente está en la línea posterior como en nuestro ejemplo. Sin embargo, si escriben varias instrucciones en una sola línea, como en el siguiente ejemplo, el punto y coma es obligatorio. Si lo virtual punto

Código JavaScript

```
sentencia_1; sentencia_2
sentencia_3
```

Compresión de scripts

Algunas secuencias de comandos están disponibles en un formato comprimido, es decir, todo el código se escribe como secuencia, no hay retornos de línea. Esto reduce considerablemente el tamaño de una secuencia de comandos y sirve para asegurar que la página se carga más rápidamente. Existen programas para "comprimir" el código Javascript. Pero si has olvidado un solo punto y coma el código comprimido no va a funcionar porque las instrucciones no están debidamente separadas. También es una de las las razones por las que siempre se pone el punto y coma al final de la instrucción.

Espacios

Javascript no es sensible a los espacios. Esto significa que puedes alinear las instrucciones que quieras, siempre que no interfiera con la secuencia de comandos. Por ejemplo, esto es correcto:

Código JavaScript

```
instruccion_1;
        instruccion_1_1;
        instruccion_1_2;
instruccion_2;      instruccion_3;
```

Sangría y presentación

La sangría en la programación informática, es una manera de estructurar el código para hacerlo más legible. Las instrucciones son priorizadas en varios niveles y espacios de usos o lengüetas para desplazar a la derecha y crear una jerarquía. Un ejemplo de código sangrado:

Código: JavaScript

```
function interruptor(elemID) {
        var elem = document.getElementById(elemID);

        if (elem.style.display == 'block') {
```

```
        elem.style.display = 'none';
        } else {
        elem.style.display = 'block';

        }
}
```

La presentación de los códigos también es importante, es como si estuvieras escribiendo una carta: no hay reglas predefinidas para escribir cartas, por lo que tendrás que hacer arreglos para organizar tu código para mostrarlo claro. En el código sangrado mostrado anteriormente, se puede ver que hay espacios para “airear” todo el código y sólo hay una declaración por línea (salvo if else, pero ya llegaremos a eso más adelante). Algunos desarrolladores escriben su código como este:

Código: JavaScript

```
function interruptor(elemID){
        var elem=document.getElementById(elemID);
        if(elem.style.display=='block'){
                elem.style.display='none';
        }else{elem.style.display='block';}
}
```

Comentarios

Los comentarios son anotaciones realizadas por el desarrollador para explicar el funcionamiento de un script, una instrucción o incluso un grupo de instrucciones. Los comentarios no interfieren con la ejecución de un script.

Hay dos tipos de comentarios: los de fin de línea y los multilínea.

Comentarios de fin de línea. Se utilizan para comentar una instrucción. Comienza con dos barras de división:

Código: JavaScript

```
sentencia_1 / / Esta es mi primera instrucción
sentencia_2;
/ / La tercera declaración es la siguiente:
sentencia_3;
```

El texto colocado en un comentario se ignora cuando se ejecuta un script, lo que significa que puedes poner un comentario, incluso en una instrucción (que, obviamente, no se ejecutará):

Código: JavaScript

```
sentencia_1 / / Esta es mi primera instrucción
sentencia_2;
/ / La tercera declaración da problemas, la cancelo temporalmente
/ / sentencia_3;
```

Comentarios multilínea. Este tipo permite saltos de línea. Un comentario multilínea comienza con / * y termina con * /:

Código: JavaScript

```
/ * Este script consta de tres pasos:
- Instrucción uno está haciendo algo
- Instrucción dos para otra cosa
- Instrucción tres que pone fin a la secuencia de comandos
* /
sentencia_1;
sentencia_2;
sentencia_3 / / Fin del script
```

Ten en cuenta que un comentario de varias líneas se pueden mostrar en una sola línea:

Código: JavaScript

```
sentencia_1 / * Esta es mi primera instrucción * /
sentencia_2;
```

Funciones

En el ejemplo de ¡Hola mundo!, Se utilizó la función alert (). Discutiremos en detalle la funciones de trabajo, en los capítulos siguientes, necesitarás saber el resumen de la sintaxis.

Una función consiste en dos partes: su nombre, seguido por un par de paréntesis (una apertura y un cierre):

Código: JavaScript

```
myFunction () / / "function"  quiere decir "función" en Inglés
```

Entre paréntesis se indican los argumentos que también se llaman parámetros. Estos contienen los valores que se pasan a la función. En el caso de ¡Hola mundo!, Son las palabras "¡Hola, mundo! " lo que se transfieren como parámetros:

Código: JavaScript

```
alert ('Hola mundo!');
```

Dónde colocar el código en la página

Los códigos JavaScript son insertados a través del elemento <script>. Este elemento tiene un atributo de tipo que se utiliza para indicar el tipo de lenguaje que vamos a utilizar. En nuestro caso, es JavaScript, pero podría ser otra cosa, como por ejemplo VBScript, aunque esto es extremadamente raro.

En HTML 4 y XHTML 1.x, el tipo de atributo es obligatorio. En contraste, en HTML5, no lo es. Esta es la razón por la que los ejemplos aquí mostrados, no incluirán este atributo. Si no estás usando HTML5, sabemos que el atributo de tipo toma como valor text / javascript, que es en realidad el tipo MIME de un código Javascript.

El tipo MIME es un identificador que describe un formato de datos. Aquí, con text / javascript, se trata de datos de texto y JavaScript.

Javascript "en la página"

Para situar el código JavaScript directamente en una página web, nada más simple, siguiendo el ejemplo de ¡Hola, mundo!: se coloca el código en el elemento <script>:

Código: HTML

```
<!DOCTYPE html>
<html>
	<head>
		<title>¡Hola Mundo!</title>
	</head>
<body>
	<script>
		alert('¡Hola Mundo!');
	</script>
</body>
</html>
```

Encuadramiento de caracteres reservados

Si utilizas el estándar HTML 4.01 y XHTML 1.x a menudo, es necesario utilizar comentarios de encuadramiento para que la página cumpla con las normas. Si por contra, como en este supuesto, se utiliza el estándar HTML5, los comentarios de encuadramiento son inútiles.

Los comentarios de encuadramiento se utilizan para aislar el código Javascript para que el Validator W3C (*World Wide Web Consortium*) no los interprete. Por ejemplo, si tu código Javascript contiene galones < y >, el validador cree que HTML no está cerrado, por lo que anularía la página. Esto no es grave en sí mismo, sino que una página sin errores, siempre es preferible.

Los comentarios de encuadramiento son como los comentarios HTML:

Código: HTML

```
<body>
<script>
        <!--
        valor_1 > valor_2;
        //-->
</script>
</body>
```

Javascript externo

Es posible, y conveniente escribir el código JavaScript en un archivo externo con la extensión. Js. Este archivo se llama desde la página web mediante el elemento <script> y su atributo src que contiene la dirección URL del archivo. js.

He aquí un pequeño ejemplo:

Código: JavaScript - contenido de ficheros hola.js

```
alert ('¡Hola mundo!');
```

Código: HTML - Página Web

```
<! DOCTYPE html>
<html>
        <head>
        <title>¡Hola mundo!!</title>
        </head>
<body>
        <script src="hola.js"></script>
```

```
</body>
</html>
```

Se supone que el archivo hola.js se encuentra en el mismo directorio que el programa en HTML.

Posición del elemento <script>

La mayoría de los cursos de Javascript, y ejemplos, muestran la necesidad de colocar el elemento <script> dentro de <head> cuando se utiliza para cargar un archivo javascript. Eso es correcto, sí, pero hay mejoras.

Una página web es leída por el navegador de forma lineal, es decir, en primer lugar lee <head>, después los elementos de <body> uno después del otro. Si se llama a un archivo JavaScript desde el principio de la carga de la página, el navegador por lo tanto cargará este archivo, y si es grande, la carga de la página se desacelerará. Esto es normal, ya que el navegador cargará el archivo antes de empezar a mostrar el contenido de la página.

Para superar este problema, es conveniente colocar los elementos <script> justo antes de cerrar <body> (algunos navegadores modernos lo hacen automáticamente) como el siguiente ejemplo:

Código: HTML

```
<!DOCTYPE html>
<html>
<head>
	<title>¡Hola Mundo!</title>
</head>
<body>
	<p>
	<!--
	Contenido de la página Web
	...
	-->
	</p>
<script>
// Un poco de código JavaScript...
	</script>
	<script src="hola.js"></script>
</body>
</html>
```

Resumen

- Las instrucciones deben estar separadas por un punto y coma.

- Un código JavaScript bien presentado es más legible y más fácil de editar.

- Es posible incluir comentarios con los caracteres / /, / * y / *.

- Los códigos Javascript son colocados en un elemento de script.

- Es posible incluir un archivo JavaScript con el atributo src del elemento <script>.

Variables

A lo largo de la lectura, descubrirás el uso de variables, los principales tipos que pueden contener y sobre todo la forma de hacer tus primeros cálculos. También se presenta la concatenación y los tipos de conversión. Y, por último, una parte importante de este capítulo trata el uso de una nueva característica que permite interactuar con el usuario.

¿Qué es una variable?

En pocas palabras, una variable es un espacio de almacenamiento en un ordenador para grabar cualquier tipo de datos como una cadena de caracteres, un valor numérico o estructuras un poco más específicas.

Declarar una variable

En primer lugar, ¿qué significa "declarar una variable" significa? Se trata simplemente de espacio de almacenamiento de reserva en memoria, nada más. Una vez que se declara la variable, puedes comenzar a almacenar datos sin problema.

Para declarar una variable, primero debes encontrar un nombre. Es importante destacar que el nombre de una variable puede contener sólo caracteres alfanuméricos, es decir, letras de la A a la Z y números del 0 al 9, guión bajo (_) y dólar ($) también son aceptados. Algo más: el nombre de la variable no puede comenzar con un número y no puede consistir únicamente de palabras clave utilizadas por Javascript. Por ejemplo, no se puede crear una variable llamada var porque encontrarás que esta palabra clave ya está en uso, sin embargo, puedes crear una variable llamada var_.

En las palabras clave utilizadas por JavaScript, se pueden llamar "palabras reservadas", simplemente porque no tienes el derecho de usarlos como nombres de variables. Encontrarás en esta página (en Inglés) todas las palabras reservadas en Javascript.

Para declarar una variable, simplemente hay que escribir la siguiente línea:

Código: JavScript

```
var miVariable;
```

JavaScript es un lenguaje sensible en las denominaciones, ten cuidado de no confundir las mayúsculas y minúsculas. En el siguiente ejemplo, tenemos tres variables diferentes declaradas:

Código: JavaScript