JAVASCRIPT : INTRODUCCIÓN

¿Qué es?

Javascript es un **lenguaje no compilado** que funciona gracias a un **intérprete de código que se encuentra en el navegador Web**. Así, no podremos ejecutar un script en este lenguaje si no es a través de un navegador o una herramienta que disponga de un intérprete.

El lenguaje fue **inventado** por **Brendan Eich** en la empresa **Netscape** Communications, que es la que desarrolló los primeros navegadores web comerciales. Apareció por primera vez en el producto de Netscape llamado Netscape Navigator 2.0.

Tradicionalmente, se venía utilizando en páginas web HTML, para realizar tareas y operaciones en el marco de la aplicación **únicamente cliente**, **sin acceso a funciones del servidor**. Javascript se ejecuta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

Los autores inicialmente lo llamaron Mocha y más tarde LiveScript pero fue rebautizado como **javascript** en un anuncio conjunto entre Sun Microsystems y Netscape, el 4 de diciembre de **1995**.

En **1997** los autores propusieron javascript para que fuera adoptado como estándar de la European Computer Manufacturers' Association ECMA. En junio de 1997 fue adoptado como un **estándar ECMA**, con el nombre de ECMAScript. Poco después también lo fue como un estándar ISO.

JScript es la implementación de ECMAScript de Microsoft, muy similar al javascript de Netscape, pero con ciertas diferencias en el modelo de objetos del navegador que hacen a ambas versiones con frecuencia incompatibles. Para evitar estas incompatibilidades, el W3C diseñó el estándar **Document Object Model** (DOM, ó Modelo de Objetos del Documento).

Incluir JavaScript en documentos HTML

La integración de JavaScript y HTML es muy flexible. Existen al menos tres formas para incluir código JavaScript en las páginas web.

Incluir JavaScript en el mismo documento HTML

El código JavaScript se encierra entre **etiquetas** `<script>` y se incluye **en cualquier parte del documento**. Aunque es correcto incluir cualquier bloque de código en cualquier zona de la página, **se recomienda definir el código JavaScript dentro de la cabecera del documento (**dentro de la etiqueta `<head>`):

```
<!DOCTYPE html>
```

```
<html>
<head>
<meta charset="utf-8" />
<title>Ejemplo de código JavaScript en el propio
documento</title> <script>
 alert("Un mensaje de prueba");
</script>
</head>
<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```

En **HTML5 no es necesario especificar ningún atributo más**, javascript es el lenguaje por defecto. En cambio, en **XHTML**, para que el documento sea válido debe incluirse el **atributo type** indicando el tipo de lenguaje de script utilizado:

```
<script type="text/javascript">
 alert("Un mensaje de prueba");
</script>
```

Este **método** se emplea cuando se define un **bloque pequeño de código** o cuando se quieren incluir instrucciones específicas en un determinado documento HTML que completen las instrucciones y funciones que se incluyen por defecto en todos los documentos del sitio web.

El principal **inconveniente** es que si se quiere hacer una **modificación** en el bloque de código, es necesario **modificar todas las páginas** que incluyen ese mismo bloque de código JavaScript.

Definir JavaScript en un archivo externo

Las instrucciones JavaScript se pueden incluir en un **archivo externo** de tipo JavaScript que los documentos HTML **enlazan mediante la etiqueta `<script>`**. Se pueden crear **todos los archivos JavaScript que sean necesarios** y cada documento HTML puede **enlazar tantos archivos JavaScript como necesite**.

Ejemplo: **Archivo codigo.js:**

```
alert("Un mensaje de prueba");
```

Documento HTML:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>Ejemplo de código JavaScript en el propio
documento</title> <script src="/js/codigo.js"></script>
</head>
<body>
```

```
<p>Un párrafo de texto.</p>
</body>
</html>
```

Este método requiere **definir el atributo src**, que es el que indica la URL correspondiente al archivo JavaScript que se quiere enlazar. **Cada etiqueta `<script>` solamente puede enlazar un único archivo**, pero en una misma página se pueden incluir tantas etiquetas `<script>` como sean necesarias.

Los archivos de tipo JavaScript son **documentos normales de texto con la extensión .js**, que se pueden crear con cualquier editor de texto.

Se considera más apropiado separar el código javascript en un archivo independiente, ya que esto tiene una serie de **ventajas** como son:

- Mayor **claridad**, se simplifica el código HTML de la página
- **Reutilización** de código, vinculando el mismo archivo javascript en distintos documentos.
- Cualquier **modificación** realizada en el archivo JavaScript se ve reflejada inmediatamente en todas las páginas HTML que lo enlazan
- Mayor **velocidad**, al descargarse el código javascript una única vez y no una vez para cada documento.

Incluir JavaScript en los elementos HTML

Este último método es el menos utilizado, ya que consiste en i**ncluir trozos de JavaScript dentro del código HTML** de la página:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>Ejemplo de código JavaScript en el propio
documento</title>
</head>
<body>
<p onclick="alert('Un mensaje de prueba')">Un párrafo de
texto.</p> </body>
</html>
```

El mayor inconveniente de este método es que **ensucia innecesariamente el código** HTML de la página y **complica el mantenimiento** del código JavaScript. En general, este método sólo se utiliza para **definir algunos eventos** y en algunos otros casos especiales.

Etiqueta noscript

Algunos **navegadores no disponen** de **soporte** completo de **JavaScript**, otros navegadores **permiten bloquearlo parcialmente** e incluso algunos usuarios **bloquean completamente** el uso de JavaScript porque creen que así navegan de forma más segura.

En estos casos, es habitual que si la página web requiere JavaScript para su correcto funcionamiento, se incluya un **mensaje** de aviso al **usuario** indicándole que debería **activar JavaScript** para disfrutar completamente de la página.

El lenguaje HTML define la etiqueta `<noscript>` para mostrar un mensaje al usuario cuando su navegador no puede ejecutar JavaScript. El siguiente código muestra un ejemplo del uso de la etiqueta `<noscript>`:

```
<body>
<script>
<!--
document.write("Hello World!")
//-->
</script>
<noscript>
 <p>Bienvenido a Mi Sitio</p>
 <p>La página que estás viendo requiere para su funcionamiento el uso de
JavaScript.
Si lo has deshabilitado intencionadamente, por favor vuelve a
activarlo.</p>
</noscript>
</body>
```

La etiqueta `<noscript>` debe incluirse en el interior de la etiqueta `<body>` (normalmente se incluye al principio de `<body>`). El mensaje que muestra `<noscript>` puede incluir cualquier elemento o etiqueta HTML en versiones distintes de HTML5. En HTML5 puede incluirse en el `<body>` o en el `<head>` , si se incluye entre la etiqueta `<head>` solo puede contener los elementos <link>, <style> y <meta>.

Pueden utilizarse los comentarios de etiqueta `<!-- //-->` para ocultar el código y que no se muestre como texto si no tiene activado el javascript.

Sintaxis

La sintaxis de un lenguaje de programación se define como el **conjunto de reglas** que deben seguirse al escribir el **código fuente** de los programas para considerarse como correctos para ese lenguaje de programación.

La sintaxis de JavaScript es muy **similar** a la de otros lenguajes de programación como **Java** y **C**. Las normas básicas que definen la sintaxis de JavaScript son las siguientes:

- **No se tienen en cuenta los espacios en blanco y las nuevas líneas**: como sucede con HTML, el intérprete de JavaScript ignora cualquier espacio en blanco sobrante, por lo que **el código se puede ordenar** de forma adecuada para **entenderlo mejor** (tabulando las líneas, añadiendo espacios, creando nuevas líneas, etc.)

- **Se distinguen las mayúsculas y minúsculas**: Si en JavaScript se intercambian mayúsculas y minúsculas el script no funciona.

- **No se define el tipo de las variables**: al crear una variable, no es necesario indicar el tipo de dato que almacenará. De esta forma, **una misma variable** puede almacenar **diferentes tipos de datos** durante la ejecución del script.

- **No es necesario terminar cada sentencia con el carácter de punto y coma** (;): en la mayoría de lenguajes de programación, es obligatorio terminar cada sentencia con el carácter ;. Aunque JavaScript no obliga a hacerlo, **es conveniente** seguir la tradición de terminar cada sentencia con el carácter del punto y coma (;).

- **Se pueden incluir comentarios**: los comentarios se utilizan para añadir información en el código fuente del programa. Aunque el contenido de los comentarios no se visualiza por pantalla, si que se envía al navegador del usuario junto con el resto del script, por lo que es necesario extremar las precauciones sobre la información incluida en los comentarios.

JavaScript define dos tipos de comentarios: los de una sola línea y los que ocupan varias líneas.

- Los **comentarios de una sola línea** se definen añadiendo **dos barras oblicuas (//) al principio de la línea**. Ejemplo:

```
// a continuación se muestra un mensaje
alert("mensaje de prueba");
```

- Los **comentarios multilínea** se definen encerrando el texto del comentario **entre los símbolos /* y */**. Ejemplo :

```
/* Los comentarios de varias líneas son muy útiles
cuando se necesita incluir bastante información
en los comentarios */
alert("mensaje de prueba");
```

Posibilidades y limitaciones

HTML5 puede ser considerado como un edificio con tres columnas: HTML, CSS y Javascript. Javascript había sido considerado hasta ahora como solo un complemento de HTML y CSS. Una de las innovaciones que ayudó a cambiar esta imagen fueron los **nuevos motores de procesamiento incorporados por las recientes versiones de los navegadores más populares.**

Estos motores fueron desarrollados para **acelerar el procesamiento del código** Javascript convirtiéndolo en código máquina y así alcanzar velocidades de ejecución similares a las aplicaciones de escritorio. Esta nueva capacidad ayudó al lenguaje a superar limitaciones previas y **confirmarlo como la mejor opción de codificación para**

la web.

Gracias a los nuevos intérpretes y motores de procesamiento como **V8 de Google**, se ha convertido en un lenguaje más poderoso e independiente. JavaScript es el responsable de la revolución de Internet en estos últimos años así como el surgimiento del nuevo mercado de dispositivos móviles.

Para aprovechar esta prometedora arquitectura, Javascript ha sido mejorado con el objetivo de lograr **mayor portabilidad e integración**. Además, incorpora interfaces de programación de aplicaciones completas (APIs) (como Web Storage, Canvas y otras) son interfaces de librarías de código incluidas en los navegadores.

En este momento, **Javascript es el lenguaje más popular y prometedor de todos**, y HTML5 lo está convirtiendo en una herramienta esencial para desarrolladores de aplicaciones web y móviles.

En cuanto a las **limitaciones**, JavaScript fue diseñado de forma que se ejecutara en un entorno muy limitado que permitiera a los **usuarios confiar en la ejecución de los scripts**.

De esta forma, los scripts de JavaScript **no pueden comunicarse con recursos que no pertenezcan al mismo dominio desde el que se descargó el script**. Los scripts **tampoco pueden cerrar ventanas que no hayan abierto esos mismos scripts**. Las ventanas que se crean no pueden ser demasiado pequeñas ni demasiado grandes ni colocarse fuera de la vista del usuario (aunque los detalles concretos dependen de cada navegador).

Además, los scripts **no pueden acceder a los archivos del ordenador del usuario** (ni en modo lectura ni en modo escritura) y **tampoco pueden leer o modificar las preferencias del navegador**.

Por último, si la ejecución de un **script dura demasiado tiempo** (por ejemplo por un error de programación) el navegador informa al usuario de que un script está consumiendo demasiados recursos y le da la **posibilidad de detener su ejecución**.

A pesar de todo, **existen alternativas** para poder saltarse algunas de las limitaciones anteriores.

El primer script

A continuación, se muestra un primer script sencillo pero completo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>El primer script</title>
<script>
 alert("Hola Mundo!");
</script>
</head>
<body>
```

```
<p>Esta página contiene el primer script</p>
</body>
</html>
```

En este ejemplo, el script se incluye como un bloque de código dentro de una página HTML. Por tanto, en primer lugar se debe crear una **página HTML correcta** que incluya la declaración del **DOCTYPE**, las secciones `<head>` y `<body>`, la etiqueta `<title>`, etc. Aunque el código del **script** se puede incluir en cualquier parte de la página, se recomienda **incluirlo en la cabecera del documento**, es decir, dentro de la etiqueta `<head>`. A continuación, el código JavaScript se debe incluir entre las etiquetas `<script>...</script>`.

Una vez definida la zona en la que se incluirá el script, se escriben todas las **sentencias que forman la aplicación**. Este primer ejemplo es tan sencillo que solamente incluye una sentencia: `alert("Hola Mundo!");`.

La instrucción `alert()` es una de las utilidades que incluye JavaScript y permite **mostrar un mensaje en la pantalla del usuario**. Si se visualiza la página web de este primer script en cualquier navegador, automáticamente se mostrará una ventana con el mensaje "Hola Mundo!".