JavaScript

Lòpez, Juan.

jl091505@ingenieria.sanmartin.edu.co

Fundación Universitaria San Martin

*Resumen*— JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.

Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java.

*Índice de Términos—JavaScript,* *break, case, catch, continue.*

# 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 XHTML, 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: al igual que sucede con la sintaxis de las etiquetas y elementos XHTML. Sin embargo, si en una página XHTML se utilizan indistintamente mayúsculas y minúsculas, la página se visualiza correctamente, siendo el único problema la no validación de la página. En cambio, 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.

## Sentencias de control

Se consigue con las sentencias if...else (bifurcación condicional) y switch (elección múltiple).

if (condición)

{

sentencias

}

else

{

sentencias alternativas

}

La expresión 'condición' debería devolver true o false.

La sentencia switch es equivalente a varios if...else anidados:

if (condicion1)

{

}

else if (condicion2)

{

}

else if (condicion3)

{

}

Su sintaxis es:

switch (expresion)

{

case valor1:

sentencias 1

....

break;

case valor2:

sentencias 2

....

break;

default:

sentencias por defecto

}

La palabra clave 'break' permite salir del bloque switch cuando se ejecutan las sentencias correspondientes al valor coincidente con 'expresion':

switch (cadena)

{

case "rojo":

case "verde":

case "azul":

tipo\_de\_color = "primario";

break;

case "negro":

tipo\_de\_color = "nulo";

break;

case "blanco":

tipo\_de\_color = "blanco";

break;

default:

tipo\_de\_color = "composición";

}

.

## Sentencias de iteracion

Bucle FOR

La sintaxis del bucle FOR es:

for ( [expresión inicial] ; [condición] ; [expresión incremental] )

{

sentencias

...

}

Que se leería como: 'Partiendo del valor proporcionado por la expresión inicial, ejecuta las sentencias contenidas en el bloque mientras se cumpla la condición. En cada nueva iteración ejecuta la expresión incremental'

El funcionamiento interno sería:

1.- Se ejecuta la expresión inicial

2.- Se evalúa la condición. Si es true continúa con el bloque de sentencias

3.- Se ejecuta el bloque de sentencias

4.- Se ejecuta la expresión incremental y se pasa al punto 2

Bucles WHILE y DO ... WHILE

Sintaxis:

WHILE DO ... WHILE

while (condición)

{

sentencias;

....

}

do

{

sentencias;

....

}

while (condición)

Las sentencias del bucle while comienzan a ejecutarse sólo si se cumple la condición inicialmente. El bucle acaba cuando la condición deja de cumplirse.

Las sentencias del bucle do... while se ejecutan al menos una vez. El bucle termina cuando condición es false.

## Manejo de excepciones

Javascript, como otros muchos lenguajes de programación, incluye soporte nativo para el manejo de excepciones.

Las excepciones, son imprevistos que ocurren durante la ejecución de un programa; anormalidades que impiden o alteran el comportamiento o flujo normal de un software.

* Throw: Este comando permite enviar al navegador un evento similar al que se produce cuando ocurre algún imprevisto o nos encontramos ante un tipo inesperado de datos. El lenguaje permite enviar todo tipo de elementos, incluyendo texto, números, valores booleanos o incluso objetos. Sin embargo, la opción más usual es enviar el objeto nativo Error:
* Try/Cath: ry … Catch corresponde a un tipo de estructura de control Javascript con la que comprobar el flujo de un programa frente a comportamientos inesperados. La diferencia entre esta estructura y la anterior es que mientras throw detiene completamente la ejecución, catch realiza una acción determinada frente a los errores para proseguir después con el flujo definido.
* Este tipo de excepciones se estructuran mediante un bloque de código que evalúa una condición previa y propone en consecuencia una ejecución predefinida y otra alternativa frente a anormalidades

## Eventos

Los eventos hacen posible que los usuarios transmitan información a los programas. JavaScript define numerosos eventos que permiten una interacción completa entre el usuario y las páginas/aplicaciones web. La pulsación de una tecla constituye un evento, así como pinchar o mover el ratón, seleccionar un elemento de un formulario, redimensionar la ventana del navegador, etc.

| **Evento** | **Descripción** | **Elementos para los que está definido** |
| --- | --- | --- |
| onblur | Deseleccionar el elemento | <button>, <input>, <label>, <select>,<textarea>, <body> |
| onchange | Deseleccionar un elemento que se ha modificado | <input>, <select>, <textarea> |
| onclick | Pinchar y soltar el ratón | Todos los elementos |
| ondblclick | Pinchar dos veces seguidas con el ratón | Todos los elementos |
| onfocus | Seleccionar un elemento | <button>, <input>, <label>, <select>,<textarea>, <body> |
| onkeydown | Pulsar una tecla (sin soltar) | Elementos de formulario y <body> |
| onkeypress | Pulsar una tecla | Elementos de formulario y <body> |
| onkeyup | Soltar una tecla pulsada | Elementos de formulario y <body> |
| onload | La página se ha cargado completamente | <body> |
| onmousedown | Pulsar (sin soltar) un botón del ratón | Todos los elementos |
| onmousemove | Mover el ratón | Todos los elementos |
| onmouseout | El ratón *"sale"* del elemento (pasa por encima de otro elemento) | Todos los elementos |
| onmouseover | El ratón *"entra"* en el elemento (pasa por encima del elemento) | Todos los elementos |
| onmouseup | Soltar el botón que estaba pulsado en el ratón | Todos los elementos |
| onreset | Inicializar el formulario (borrar todos sus datos) | <form> |
| onresize | Se ha modificado el tamaño de la ventana del navegador | <body> |
| onselect | Seleccionar un texto | <input>, <textarea> |
| onsubmit | Enviar el formulario | <form> |
| onunload | Se abandona la página (por ejemplo al cerrar el navegador) | <body> |

## Métodos de manipulación del DOOM

Para poder utilizar las funciones de manipulación de elementos, DOM tiene que transformar internamente el archivo XHTML original en una estructura más fácil de manipular. Esta estructura es una estructura en forma de árbol de nodos, donde cada elemento es un nodo del árbol.

Si JavaScript es el primer lenguaje de programación que utilizáis puede que os cueste un poco entender el concepto de árbol de nodos, así que vamos a ver un ejemplo sencillo de un documento XHTML y de árbol de nodos, extraído del libro de JavaScript de Javier Eguíluz.



Utilizando las funciones de DOM podemos añadir, eliminar y modificar cualquier nodo de la página, ya sea para añadirle nuevos atributos, nuevos estilos CSS o algún que otro manejador de eventos. Lo malo de todo esto es que las especificaciones de DOM que realiza cada navegador incluye algunas diferencias que nos complican las cosas para crear aplicaciones que se ejecuten bien en diferentes navegadores. Por eso nacieron algunos frameworks JavaScript, como Prototype o jQuery, que proporcionan una solución cross-browser.

## Notacion JSON y su uso

JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituído por dos estructuras:

Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.

Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o sequencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

referencias

1. <http://librosweb.es/javascript/capitulo_1.html>
2. http://www.json.org/json-es.html.