APRENDERAPROGRAMAR.COM

USE STRICT JAVASCRIPT. QUÉ ES STRICT MODE (MODO ESTRICTO). WITH. THE GOOD PARTS. HACIA NUEVAS VERSIONES. (CU01189E)

Sección: Cursos
Categoría: Tutorial básico del programador web: JavaScript desde cero

Fecha revisión: 2029

Resumen: Entrega nº89 del Tutorial básico "JavaScript desde cero".

Autor: César Krall

STRICT MODE

El lenguaje JavaScript ha sido denostado por algunos por considerarlo poco consistente y alabado por otros por su gran potencia. Se suele citar el libro "JavaScript: The Good Parts" (JavaScript: las partes buenas) como referente para ilustrar que JavaScript tiene cosas buenas y otras cosas que no son buenas y por tanto supuestamente no se deberían usar.

Nosotros consideramos que las personas que adquieran un conocimiento avanzado de JavaScript deben guiarse por su propio criterio y decidir cómo usarlo. Por el contrario, las personas que sólo tienen un conocimiento superficial tienen que guiarse por lo que hacen o proponen otros. Te animamos a que progresivamente vayas conformando tu propio criterio.

DIRECTIVA USE STRICT

La directiva use strict es una directiva que no supone una instrucción de código, sino que indica el modo en que el navegador debe ejecutar el código JavaScript. Podríamos hablar de dos modos de ejecución JavaScript: el <<normal mode>>, que es el que hemos estudiado hasta ahora, y el <<strict mode>>, que vamos a explicar.

Declarar que se use strict mode supone algunos cambios en cuanto al código que admite o no admite el navegador. Por ejemplo en strict mode es obligatoria la declaración de variables, mientras que en el modo normal no es necesario declarar una variable para usarla.

Para indicar que el código debe ser considerado en modo estricto se escribe lo siguiente:

```
'use strict';
```

Esta directiva la escribiremos normalmente al comienzo de un fichero, en cuyo caso afectará a todo el código y todas las funciones existentes dentro de él, o bien al comienzo de una función, en cuyo caso afectará sólo al código en el ámbito de dicha función.

Para situarnos, activa la consola del navegador y procede a hacer lo siguiente: crea un archivo html con el siguiente contenido y guárdalo con el nombre de archivo sinstrict.html:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html><head><title>Ejemplo aprenderaprogramar.com</title><meta charset="utf-8">
<script type="text/javascript">
function ejemplo() { nombre = 'Carlos'; alert('Soy '+nombre); }
</script></head>
<body><div id="cabecera"><h2>Cursos aprenderaprogramar.com</h2><h3>Ejemplos JavaScript</h3></div>
<div style="color:blue;" id ="pulsador" onclick="ejemplo()"> Probar </div>
</body></html>
```

El resultado esperado es que no haya mensajes de error y que al pulsar en “Probar” por pantalla se muestre el mensaje <<Soy Carlos>>.

Crea ahora un archivo html con el siguiente contenido y guárdalo con el nombre de archivo constrict.html:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Ejemplo aprenderaprogramar.com</title>
<meta charset="utf-8">
<script type="text/javascript">
function ejemplo() {
'use strict';
nombre = 'Carlos';
alert('Soy '+nombre);
}
</script>
</head>
<body>
<div id="cabecera">
<h2>Cursos aprenderaprogramar.com</h2>
<h3>Ejemplos JavaScript</h3>
</div>
<div style="color:blue;" id ="pulsador" onclick="ejemplo()"> Probar </div>
</body>
</html>
```

El resultado esperado es que cuando se pulse sobre “Probar” no se muestre ningún mensaje y podremos observar un error en consola del tipo << ReferenceError: assignment to undeclared variable nombre>>.

Aquí vemos cómo el strict mode está afectando a qué hace el intérprete del navegador. Sin strict mode, asumen que cualquier variable no declarada debe considerarse creada implícitamente. Con strict mode, asume que una variable no declarada generará un error.

Ten en cuenta que algunos navegadores, especialmente los más antiguos, no reconocen la directiva strict mode, y en estos navegadores el resultado es que no tiene ningún efecto (como si no existiera).

¿Por qué apareció el strict mode? En versiones antiguas de JavaScript no existía el strict mode. Este surgió debido a que parte de la comunidad de programadores se quejaba porque JavaScript era poco seguro a la hora de programar y se consideraba que sería adecuado introducir restricciones que forzaran a que el código debiera cumplir mayores requisitos para garantizar su coherencia. Por ejemplo, es habitual considerar que es una mala práctica no declarar las variables antes de usarlas porque puede inducir a errores (por ejemplo, si tenemos una variable cuyo nombre es <<aceptado>> y aparece otra variable cuyo nombre es <<aceptada>>, si ambas están declaradas se entiende que son variables distintas. Pero si no están declaradas, ¿son variables distintas o podría tratarse de un error al escribir el nombre de la variable?).

Además de no permitir el uso de variables declaradas, hay otras prácticas de programación que son admitidas por JavaScript clásico pero que no son admitidas en strict mode.

Citaremos algunas:

Práctica admitida en modo normal pero no admitida en strict mode	Ejemplo
Uso de variables no declaradas	nombre = 'juan';
Borrar una variable u objeto usando delete	delete nombre;
Definir una propiedad dos veces	var x = {persona:'juan', persona:'juan'};
Nombres de parámetros duplicados	function saludar(persona, persona) {};
Usar eval como nombre de variable	eval = 'aprobado';
En una función si no se conoce this es el objeto global window	En una función si no se conoce this es undefined
Otras	Hay más restricciones que impone el strict mode, relacionadas con evitar el uso de sintaxis no adecuada, instrucciones poco adecuadas, prácticas inseguras y mejora de la seguridad para el usuario.

En resumen, el strict mode supone que muchos fallos que JavaScript "se tragaba", dejen de ser aceptados y aparezcan como errores no admitidos (que detendrán la ejecución del script o deberán ser capturados).

STRICT MODE Y LAS NUEVAS VERSIONES DE JAVASCRIPT

Strict mode trata de hacer más segura y fiable la programación JavaScript, y elimina algunas de las denominadas "Bad parts" o partes malas del lenguaje.

La aparición de strict mode posiblemente trata de ser una transición entre la situación y el JavaScript aceptado por navegadores antiguos y lo que serán las versiones del futuro de JavaScript, donde todo o parte de las características del strict mode pasarán a ser características intrínsecas de JavaScript.

Muchos programadores y librerías de JavaScript trabajan con strict mode, y otros muchos no lo hacen.

Programar bajo strict mode puede considerarse una buena práctica. Sin embargo, no es posible o no es aconsejable tomar scripts, páginas web o aplicaciones web y aplicarles el strict mode porque pueden dejar de funcionar en algunos navegadores, especialmente los más antiguos.

Por otro lado, usar strict mode puede inducir que el código tenga algunos comportamientos diferentes a lo esperado. También puede haber problemas si se combinan partes de código o archivos que usan strict mode con otros que no lo usan. Por ello, en caso de usar strict mode se recomienda que se hagan

pruebas en navegadores que no soportan strict mode y en navegadores que sí lo soportan, para evitar problemas de compatibilidad, es decir, que los scripts funcionen bien en algunos navegadores pero no en otros.

Usar strict mode no nos convertirá en mejores programadores, pero posiblemente nos ayudará a que no se escapen errores que podrían escaparse sin usarlo.

WITH JAVASCRIPT

Hay una cláusula o sentencia de manipulación de objetos JavaScript de la que no hemos hablado: with. Se usa de la siguiente manera:

```
with (expresión que crea un contexto de referencia) {
  sentencias donde el contexto de referencia por defecto es el indicado
}
```

Una aplicación de with podría ser aplicar propiedades a un objeto, por ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Ejemplo aprenderaprogramar.com</title>
<meta charset="utf-8">
<script type="text/javascript">
function ejemplo() {
with(document.getElementById('pulsador').style) {
   backgroundColor = 'yellow';
   color = 'black';
   width = '200px';
   padding = '20px';
            fontSize = '32px';
}
 }
</script>
</head>
<body>
<div id="cabecera"><h2>Cursos aprenderaprogramar.com</h2><h3>Ejemplos JavaScript</h3></div>
<div style="color:blue;" id ="pulsador" onclick="ejemplo()"> Probar </div>
</body>
</html>
```

En JavaScript el uso de with se considera una mala práctica. Los motivos para ello son que puede dar lugar a comportamientos inesperados, problemas de rendimiento y problemas de seguridad.

Un problema potencial es que si se trata de invocar una propiedad que no existe en el contexto del with, en lugar de crearse una nueva propiedad, se crea una variable global completamente nueva e independiente. Esto puede ser difícil de detectar y dar lugar a problemas.

Ejemplo:

```
function ejemplo() {
with(document.getElementById('pulsador').style) {
   backgroundColor = 'yellow';
   color = 'black';
   width = '200px';
   padding = '20px';
   font_Size = '32px';
}
}
```

Aquí parece que font_Size aluda al tamaño de fuente del nodo con id <<pulsador>>, o al menos que sería una propiedad del objeto asociado. Sin embargo, no es así. Al llevar el guión bajo en font_Size no se reconoce la propiedad y se crea una variable global nueva.

Otro problema del with es que obliga al intérprete a buscar primero entre todas las propiedades del objeto referenciado, y si no se encuentra el nombre utilizado, entonces buscarlo en otro contexto o crear algo nuevo. Esto lleva a que trabajar con with conlleve un mal rendimiento. Especialmente en bucles dentro de with, el rendimiento o velocidad de ejecución se ve drásticamente reducido.

Por estos y otros motivos with ha pasado a considerarse no permitido en strict mode (donde obtendremos un error de tipo <<SyntaxError: strict mode code may not contain 'with' statements>>) y se considera que terminará por desaparecer (o ser replanteado de otra manera).

EJERCICIO 1

Dado este fragmento de código. Revísalo y responde a las siguientes preguntas:

```
var persona1 = {};
Object.defineProperty(persona1, "edad", { value: 42, writable: false });
persona1.edad = 19;
```

a) Explica paso a paso el significado de este código (busca información en internet si te es preciso).

b) Crea un pequeño script donde se ejecute este código y se muestre un mensaje por pantalla informando del valor de la edad. Activa la consola para comprobar si aparece algún error.

c) Crea el mismo script pero usando strict mode. Activa la consola y comprueba si aparece algún error. ¿Qué diferencias observas entre la ejecución con strict mode y sin strict mode? ¿Qué explicación le darías a estas diferencias? ¿Crees que sería positivo que este código se escribiera en strict mode o no? ¿Por qué?

Para comprobar si tus respuestas y código son correctos puedes consultar en los foros aprenderaprogramar.com.

EJERCICIO 2

Crea y ejecuta un script que use with y que esté:

a) En modo normal. ¿Cuál es el código y cuál es el resultado que obtienes?

b) En strict mode. ¿Cuál es el código y cuál es el resultado que obtienes?

Para comprobar si tus respuestas y código son correctos puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU01190E

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente: http://aprenderaprogramar.com/index.php?option=com_content&view=category&id=78&Itemid=206