Universitat Oberta de Catalunya

Aula

M4.250 - Introducción a la programación en JavaScript I aula 1

Números, fechas y texto

Inicio: **10/04/20**

Fin: **21/04/20**

Descripción

En este apartado trabajaremos dos temas distintos:

- Números y fechas
- Cadenas de texto

Veréis que las traducciones del material de Mozilla empiezan a tener algunas lagunas. Sin embargo si algún punto queda poco claro esta pequeña guía debería solucionar el problema.

Números y fechas

La mayoría de lenguajes de programación tienen al menos dos tipos de datos numéricos: reales y enteros. JavaScript solo tiene un tipo de datos numéricos que equivaldría a tener solo números reales. Lo que sí podemos tener en JavaScript son números en diferentes bases:

- Decimales
- Binarios, en cuyo caso escribiremos el número empezando por 0b o 0B. Lo que venga a continuación de la letra será el número binario.
- Octales, en cuyo caso escribiremos el número empezando por un 0o o 0O. ¡Ojo!, dependiendo de la versión de ECMAScript que usemos puede o no funcionar el usar un solo 0 antes del número octal. Cómo lo habitual será usar EMACScript 6, mejor acostumbrarnos a usar 0o o 0O antes de escribir un número octal.
- Hexadecimales, que los escribiremos empezando por 0x o 0X.

Hay que decir que JavaScript hace la conversión directa a decimal de los números en otras bases. Así, si escribimos:

```
var num = 0b00010;
```

Lo que hará JavaScript es guardar un 2 en la variable **num** .

A continuación tenéis algunos ejemplos:

```
var b2 = 0b10; // Guarda un 2
var b5 = 0B101; // Guarda un 5

console.log(b2+b5);

var o10 = 0o12; // Guarda un 10
var o16 = 0O20; // Guarda un 16

console.log(o10+o16);

var x10 = 0xA;
var x20 = 0x14;

console.log(x10+x20);
console.log(b2+x10);
```

Podéis ver la ejecución de estos ejemplos en esta página: https://goo.gl/GmzBmA

El objeto Number

No hay mucho que explicar a parte de lo que pone en el material de Mozilla. Tan solo explicar qué es un "Safe Integer". Antes de explicar qué es, comentar el problema que requiere hacer esta comprobación.

En JavaScript los números se guardan en un espacio de 64 bits. Eso permite guardar con precisión números que están entre $-(2^{53}-1)$ y $2^{53}-1$. Sin embargo podemos usar números más grandes, por ejemplo 10^{200}. Sin embargo, para números que están fuera del rango de números que puede guardar con precisión JavaScript hace una aproximación con lo que el número que retorne será aproximado. Así:

```
2**53 + 0 = 9007199254740992

2**53 + 1 = 9007199254740992

2**53 + 2 = 9007199254740994

2**53 + 3 = 9007199254740996

2**53 + 4 = 9007199254740996
```

Esos números están calculados con este script:

https://codepen.io/ccasadom/pen/zeXEjG?editors=0010

Así un “Safe Integer” será un número del que estamos seguros de que será correcto, no una aproximación. Se puede encontrar más información aquí:

https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Number/isSafeInteger

El objeto Math

Para hacer operaciones que no podemos hacer con los operadores normales usaremos el objeto Math. Con el objeto Math podemos calcular senos, cosenos, logaritmos, máximos y mínimos y generar números aleatorios, entre otros. En el material de Mozilla las diferentes operaciones están enlazadas a páginas donde se encontrarán más explicaciones y ejemplos.

Fechas

En JavaScript existe un objeto **Date** que nos permitirá trabajar con fechas.

Para crear una variable que almacene fechas deberemos usar esta sintaxis:

var variable = new Date([parámetros]); // (los parámetros son opcionales)

En el material de Mozilla está bien explicado y con buenos ejemplos todo lo que se puede hacer con las fechas.

Texto
Métodos de String
Cadenas de plantillas multilínea
Internacionalización

Cuando hablemos de texto en cualquier lenguaje de programación hablaremos de Cadenas de caracteres (o tan solo cadenas) o Strings.
Las cadenas de caracteres las escribimos poniéndolas entre comillas simples o dobles, pero las mismas al empezar y al acabar, no podemos mezclar comillas.

Así, son cadenas correctas:

- 'Hola'
- "Adiós"
- "Sí, la casa era 'bonita' y espaciosa."

- 'Sí, la casa era “bonita” y espaciosa.'

Pero serían incorrectas:

- ‘Hola"
- "Adiós

Se usan para escribir caracteres especiales que no están disponibles en nuestro teclado. La opción que más nos conviene conocer es la de las secuencias de escape del código Unicode puesto que es la que nos permitirá acceder fácilmente a cualquier carácter Unicode. Si queréis pasar un rato viendo los códigos Unicode disponibles podéis visitar esta página: https://unicode-table.com

El objeto String

Igual que con los números teníamos el objeto Number que nos facilita hacer algunas operaciones y conversiones, para las cadenas de caracteres tenemos el objeto String, que, de nuevo, nos facilitará algunas operaciones para el tratamiento de cadenas. Sin embargo y de nuevo como con el objeto Number, no crearemos específicamente objetos String. Crearemos variables de tipo cadena y cuando sea necesario, JavaScript creará un objeto temporal para tratar la cadena y luego lo borrará.

Veámoslo con un ejemplo:

```
var o = new String("Hola"); //Esto crea un objeto String
var c = "Adiós"; //Esto crea una cadena.
```

Aunque son dos cosas diferentes, en la práctica JavaScript se encargará de que para nosotros no haya diferencia.

Así, podemos usar la propiedad length que nos dice el tamaño de un objeto String. Sin embargo podremos usar esa propiedad también con una cadena:

```
console.log(o.length); // Escribe 4 en la consola
console.log(c.length); // Escribe 5 en la consola
```

Para acceder a un determinado carácter de una cadena lo podremos hacer mediante su posición (empezando por la posición 0):

```
console.log(c[2]); //Escribe i en la consola
```

Sin embargo:

- A diferencia de lo que veremos en los arrays, no podemos modificar una posición de una cadena. No podemos hacer c[2] = "e"; para que c guarde “Adeós”, no funciona.
- Los datos se guardan en formato UTF-8 . Así que si tenemos un carácter UTF-16 , los resultados pueden ser inesperados, porque se guardará ocupando dos espacios en vez de uno. Por eso es mejor usar los métodos del objeto String para obtener caracteres.

Para aclarar mejor este caso, podéis ver un ejemplo en el vídeo que acompaña a esta guía. El código usado en el vídeo está disponible aquí: https://codepen.io/ccasadom/pen/aMzbWP?editors=0011

Métodos de String

En la documentación de Mozilla podéis ver la relación de los métodos disponibles para String. Todos los métodos están enlazados a una página con su correspondiente explicación. Os recomendamos probarlos todos menos los de expresiones regulares. Para alguno de los ejemplos os irá bien tener presente la tabla de códigos Unicode que enlazamos antes.

Cadenas de plantillas multilínea

De todo lo que se explica en este apartado, nos podemos quedar con lo siguiente: Si queremos guardar en una variable una cadena con dos líneas, podemos escribirla directamente en dos líneas usando el acento grave como delimitador.

Ejemplo:

```
cadena = `Esto es una cadena
escrita en dos líneas`;
```

Nota : console.log(cadena) nos mostrará las dos líneas. Sin embargo document.write(cadena), nos mostrará una sola línea. ¿Por qué?

Por otra parte, podemos incluir datos evaluados usando el símbolo $

Ejemplo:

```
var nombre = "Carlos";
var cadena = `Hola ${nombre}`;
console.log(cadena); //Escribirá Hola Carlos
```

Internacionalización

Esta parte del material de Mozilla no está traducida pero tampoco es difícil de entender. Para comprender mejor el funcionamiento de estas funciones, os recomendamos probar estos ejemplos:

```
var fecha = new Date();
var options = { year: "2-digit", month: "2-digit", day: "2-digit",
hour: "2-digit", minute: "2-digit", timeZoneName: "short" };
var americanDateTime = new Intl.DateTimeFormat("en-US", options).format;
var spanishDateTime = new Intl.DateTimeFormat("es-ES").format;
console.log(americanDateTime(fecha)); // 07/16/14, 5:00 PM PDT
document.write(americanDateTime(fecha));
document.write("
");
document.write(spanishDateTime(fecha));
```

Probad a cambiar las opciones, o eliminarlas de la fecha americana.

```
var gasPrice = new Intl.NumberFormat("en-US",
{ style: "currency", currency: "USD",
minimumFractionDigits: 3 });
console.log(gasPrice.format(5.259)); // $5.259
var precio = new Intl.NumberFormat("es-ES",
{ style: "currency", currency: "EUR",
minimumFractionDigits: 2 });
console.log(precio.format(5.259)); // 5,26 €
```

Recursos de aprendizaje

Fuentes de información