

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

MF0491_3



Realizado por Manuel Macías

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Variables:

Es el nombre genérico que se le da a pequeños espacios de memoria donde guardas una información determinada, de forma muy similar a las incógnitas en matemáticas. Un programa puede tener muchas variables, y cada una de ellas tendrá un nombre, un valor y un tipo de dato.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Declarando una variable en JavaScript

```
var variable = 2;
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Usando el contenido de una variable

```
var variable = 2;  
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU ÁMBITO

Asignando un nuevo valor a la variable

```
var variable = 2;  
variable = 3;  
console.log(variable);
```

Asignando el valor de una variable a otra variable

```
var variable = 2;  
var variable2 = variable;  
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Realizando una operación con el valor almacenado en una variable

```
var variable = 2;
```

```
variable = variable + 1; // 3
```

```
console.log(variable);
```

Realizando operaciones complejas

```
var variable = 2;
```

```
variable = variable * 25 + 3; // 53
```

```
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Controlando la precedencia en la operación

```
var variable = 2;
```

```
variable = variable * (25 + 3); // 56
```

```
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Incrementando el valor de una variable

```
var variable = 0;  
variable ++;  
console.log(variable); // 1
```

Incrementando el valor de una variable en un valor específico

```
var variable = 0;  
variable += 5;  
console.log(variable); // 5
```


PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Asignando una cadena de caracteres a una variable

```
var variable = "Hola Mundo!";
```

```
console.log(variable);
```

Concatenando texto

```
var variable = "Mi nombre es ";
```

```
variable = variable + "Juan";
```

```
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Agregando texto al comienzo del valor

```
var variable = "Juan";
```

```
variable = "Mi nombre es " + variable;
```

```
console.log(variable);
```

Concatenando texto con números

```
var variable = "El número es " + 3;
```

```
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Generando nuevas líneas de texto

```
var variable = "Felicidad no es hacer lo que uno quiere\r\n";  
variable = variable + "sino querer lo que uno hace."  
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Declarando una variable Booleana

```
var variable = true;  
console.log(variable);
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU ÁMBITO

Una de las características que llegaron con ES6 es la adición de **let** y **const**, que se pueden utilizar para la declaración de variables.

Var

Antes de la llegada de ES6, las declaraciones **var** eran las que mandaban. Sin embargo, hay problemas asociados a las variables declaradas con **var**.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Ámbito de var

El ámbito, significa esencialmente dónde están disponibles estas variables para su uso. Las declaraciones **var** tienen un ámbito global o un ámbito de función/local.

El ámbito es global cuando una variable **var** se declara fuera de una función. Esto significa que cualquier variable que se declare con **var** fuera de una función está disponible para su uso.

var tiene un ámbito local cuando se declara dentro de una función. Esto significa que está disponible y solo se puede acceder a ella dentro de esa función.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU ÁMBITO

```
var saludar = "hey, hola";  
  
function nuevaFuncion() {  
    var hola = "hola";  
  
}
```

Aquí, **saludar** tiene un ámbito global porque existe fuera de la función mientras que **hola** tiene un ámbito local. Así que no podemos acceder a la variable **hola** fuera de la función.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Las variables con **var** se pueden volver a declarar y modificar.

Esto significa que podemos hacer esto dentro del mismo ámbito y no obtendremos un error.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Hoisting de var

Hoisting es un mecanismo de JavaScript en el que las variables y declaraciones de funciones se mueven a la parte superior de su ámbito antes de la ejecución del código. Esto significa que si hacemos esto:

```
console.log (saludar);
```

```
var saludar = "dice hola"
```

se interpreta así:

```
var saludar;
```

```
console.log(saludar); // saludar is undefined
```

```
saludar = "dice hola"
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

let

let es ahora preferible para la declaración de variables. Ya que es una mejora de las declaraciones con **var**. Y también resuelve los problemas con **var**.

let tiene un ámbito de bloque

Un bloque es un trozo de código delimitado por **{}**.

Así que una variable declarada en un bloque con **let** solo está disponible para su uso dentro de ese bloque.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

```
let saludar = "dice Hola";
```

```
let tiempos = 4;
```

```
if (tiempos > 3) {
```

```
    let hola = "dice Hola tambien";
```

```
    console.log(hola); // "dice Hola tambien"
```

```
}
```

```
console.log(hola) // hola is not defined
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

let puede modificarse pero no volver a declararse.

Al igual que **var**, una variable declarada con **let** puede ser actualizada dentro de su ámbito. A diferencia de **var**, una variable **let** no puede ser re-declarada dentro de su ámbito.

```
let saludar = "dice Hola";  
saludar = "dice Hola tambien";
```

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU ÁMBITO

Sin embargo, si la misma variable se define en diferentes ámbitos, no habrá ningún error:

```
let saludar = "dice Hola";  
if (true) {  
    let saludar = "dice Hola tambien";  
    console.log(saludar); // "dice Hola tambien"  
}  
console.log(saludar); // "dice Hola"
```

¿Por qué no hay ningún error? Esto se debe a que ambas instancias son tratadas como variables diferentes, ya que tienen ámbitos diferentes.

Este hecho hace que **let** sea una mejor opción que **var**. Cuando se utiliza **let**, no hay que preocuparse de si se ha utilizado un nombre para una variable antes, puesto que una variable solo existe dentro de su ámbito.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Hoisting de let

Al igual que **var**, las declaraciones **let** se elevan a la parte superior. A diferencia de **var** que se inicializa como **undefined**, la palabra clave **let** no se inicializa. Sí que si intentas usar una variable **let** antes de declararla, obtendrás un **Reference Error**.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

const

Las variables declaradas con **const** mantienen valores constantes.
Las declaraciones **const** tiene similitudes con las declaraciones **let**.

Las declaraciones **const** tienen un ámbito de bloque

Al igual que las declaraciones **let**, solamente se puede acceder a las declaraciones **const** dentro del bloque en el que fueron declaradas.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

const no puede modificarse ni volver a declararse

Esto significa que el valor de una variable declarada con **const** es el mismo dentro de su ámbito. No se puede actualizar ni volver a declarar. Así que si declaramos una variable con **const**, no podemos hacer esto:

```
const saludar = "dice Hola";
```

```
saludar = "dice Hola tambien";// error: Assignment to  
constant variable.
```


PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU
ÁMBITO

Hoisting de const

Al igual que **let**, las declaraciones **const** se elevan a la parte superior, pero no se inicializan.

PROGRAMACIÓN WEB EN EL ENTORNO CLIENTE

VARIABLES SU ÁMBITO

En Resumen:

- Las declaraciones **var** tienen un ámbito global o un ámbito función/local, mientras que **let** y **const** tienen un ámbito de bloque.
- Las variables **var** pueden ser modificadas y redeclaradas dentro de su ámbito; las variables **let** pueden ser modificadas, pero no redeclaradas; las variables **const** no pueden ser modificadas ni redeclaradas.
- Todas ellas se elevan a la parte superior de su ámbito. Pero mientras que las variables **var** se inicializan con **undefined**, **let** y **const** no se inicializan.
- Mientras que **var** y **let** pueden ser declaradas sin ser inicializadas, **const** debe ser inicializada durante la declaración.



Programación web en el entorno cliente

GRACIAS

MANUEL MACÍAS

TUTORIAS@MANUELMACIAS.ES



PROGRAMACIÓN JAVASCRIPT