**DECLARACION DE VARIABLES EN JAVA SCRIPT**

La palabra reservada var solamente se debe indicar al definir por primera vez la variable, lo que se denomina **declarar** una variable. Cuando se utilizan las variables en el resto de instrucciones del script, solamente es necesario indicar su nombre

El nombre de una variable también se conoce como **identificador** y debe cumplir las siguientes normas:

* Sólo puede estar formado por letras, números y los símbolos $ (dólar) y \_ (guión bajo).
* El primer carácter no puede ser un número.

Por tanto, las siguientes variables tienen nombres correctos:

var $numero1;

var \_$letra;

var $$$otroNumero;

var $\_a\_\_$4;

Sin embargo, las siguientes variables tienen identificadores incorrectos:

var 1numero; // Empieza por un número

var numero;1\_123; // Contiene un carácter ";"

**Tipos de variables**

**Numéricas:** Se utilizan para almacenar valores numéricos enteros (llamados integer en inglés) o decimales (llamados float en inglés). En este caso, el valor se asigna indicando directamente el número entero o decimal. Los números decimales utilizan el carácter . (punto) en vez de , (coma) para separar la parte entera y la parte decimal:

var iva = 16; // variable tipo entero

var total = 234.65; // variable tipo decimal

Cadenas de texto: Se utilizan para almacenar caracteres, palabras y/o frases de texto. Para asignar el valor a la variable, se encierra el valor entre comillas dobles o simples, para delimitar su comienzo y su final:

var mensaje = "Bienvenido a nuestro sitio web";

var nombreProducto = 'Producto ABC';

var letraSeleccionada = 'c';

| **Si se quiere incluir...** | **Se debe incluir...** |
| --- | --- |
| Una nueva línea | \n |
| Un tabulador | \t |
| Una comilla simple | \' |
| Una comilla doble | \" |
| Una barra inclinada | \\ |

De esta forma, el ejemplo anterior que contenía comillas simples y dobles dentro del texto se puede rehacer de la siguiente forma:

var texto1 = 'Una frase con \'comillas simples\' dentro';

var texto2 = "Una frase con \"comillas dobles\" dentro";

### Arrays

var dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"];

var nombre\_array = [valor1, valor2, ..., valorN];

para acceder a los valores:

var diaSeleccionado = dias[0]; // diaSeleccionado = "Lunes"

var otroDia = dias[5]; // otroDia = "Sábado"

**Booleanos**

Las variables de tipo boolean o booleano también se conocen con el nombre de variables de tipo lógico.

Los únicos valores que pueden almacenar estas variables son true y false, por lo que no pueden utilizarse los valores verdadero y falso. A continuación se muestra un par de variables de tipo booleano:

var clienteRegistrado = false;

var ivaIncluido = true;

# Operadores

### Asignación

var numero1 = 3;

### Incremento y decremento

Estos dos operadores solamente son válidos para las variables numéricas y se utilizan para incrementar o decrementar en una unidad el valor de una variable.

Ejemplo:

var numero = 5;

++numero;

alert(numero); // numero = 6

El operador de incremento se indica mediante el prefijo ++ en el nombre de la variable. El resultado es que el valor de esa variable se incrementa en una unidad. Por tanto, el anterior ejemplo es equivalente a:

var numero = 5;

numero = numero + 1;

alert(numero); // numero = 6

### Lógicos

Los operadores lógicos son imprescindibles para realizar aplicaciones complejas, ya que se utilizan para tomar decisiones sobre las instrucciones que debería ejecutar el programa en función de ciertas condiciones.

#### Negación

Uno de los operadores lógicos más utilizados es el de la negación. Se utiliza para obtener el valor contrario al valor de la variable:

var visible = true;

alert(!visible); // Muestra "false" y no "true"

Si la variable original es de tipo *booleano*, es muy sencillo obtener su negación. Sin embargo, ¿qué sucede cuando la variable es un número o una cadena de texto? Para obtener la negación en este tipo de variables, se realiza en primer lugar su conversión a un valor *booleano*:

* Si la variable contiene un número, se transforma en false si vale 0 y en true para cualquier otro número (positivo o negativo, decimal o entero).
* Si la variable contiene una cadena de texto, se transforma en false si la cadena es vacía ("") y en true en cualquier otro caso.
* var cantidad = 0;
* vacio = !cantidad; // vacio = true
* cantidad = 2;
* vacio = !cantidad; // vacio = false
* var mensaje = "";
* mensajeVacio = !mensaje; // mensajeVacio = true
* mensaje = "Bienvenido";
* mensajeVacio = !mensaje; // mensajeVacio = false

#### AND

La operación lógica AND obtiene su resultado combinando dos valores booleanos. El operador se indica mediante el símbolo && y su resultado solamente es true si los dos operandos son true:

var valor1 = true;

var valor2 = false;

resultado = valor1 && valor2; // resultado = false

valor1 = true;

valor2 = true;

resultado = valor1 && valor2; // resultado = true

#### OR

La operación lógica OR también combina dos valores booleanos. El operador se indica mediante el símbolo || y su resultado es true si alguno de los dos operandos es true:

var valor1 = true;

var valor2 = false;

resultado = valor1 || valor2; // resultado = true

valor1 = false;

valor2 = false;

resultado = valor1 || valor2; // resultado = false

### Matemáticos

JavaScript permite realizar manipulaciones matemáticas sobre el valor de las variables numéricas. Los operadores definidos son: suma (+), resta (-), multiplicación (\*) y división (/). Ejemplo:

var numero1 = 10;

var numero2 = 5;

resultado = numero1 / numero2; // resultado = 2

resultado = 3 + numero1; // resultado = 13

resultado = numero2 – 4; // resultado = 1

resultado = numero1 \* numero 2; // resultado = 50

Además de los cuatro operadores básicos, JavaScript define otro operador matemático que no es sencillo de entender cuando se estudia por primera vez, pero que es muy útil en algunas ocasiones.

Se trata del operador "módulo", que calcula el resto de la división entera de dos números. Si se divide por ejemplo 10 y 5, la división es exacta y da un resultado de 2. El resto de esa división es 0, por lo que módulo de 10 y 5 es igual a 0.

Sin embargo, si se divide 9 y 5, la división no es exacta, el resultado es 1 y el resto 4, por lo quemódulo de 9 y 5 es igual a 4.

El operador módulo en JavaScript se indica mediante el símbolo %, que no debe confundirse con el cálculo del porcentaje:

var numero1 = 10;

var numero2 = 5;

resultado = numero1 % numero2; // resultado = 0

numero1 = 9;

numero2 = 5;

resultado = numero1 % numero2; // resultado = 4

Los operadores matemáticos también se pueden combinar con el operador de asignación para abreviar su notación:

var numero1 = 5;

numero1 += 3; // numero1 = numero1 + 3 = 8

numero1 -= 1; // numero1 = numero1 - 1 = 4

numero1 \*= 2; // numero1 = numero1 \* 2 = 10

numero1 /= 5; // numero1 = numero1 / 5 = 1

numero1 %= 4; // numero1 = numero1 % 4 = 1

### Relacionales

Los operadores relacionales definidos por JavaScript son idénticos a los que definen las matemáticas: mayor que (>), menor que (<), mayor o igual (>=), menor o igual (<=), igual que (==) y distinto de (!=).

var numero1 = 3;

var numero2 = 5;

resultado = numero1 > numero2; // resultado = false

resultado = numero1 < numero2; // resultado = true

numero1 = 5;

numero2 = 5;

resultado = numero1 >= numero2; // resultado = true

resultado = numero1 <= numero2; // resultado = true

resultado = numero1 == numero2; // resultado = true

resultado = numero1 != numero2; // resultado = false

Se debe tener especial cuidado con el operador de igualdad (==), ya que es el origen de la mayoría de errores de programación, incluso para los usuarios que ya tienen cierta experiencia desarrollando scripts. El operador == se utiliza para comparar el valor de dos variables, por lo que es muy diferente del operador =, que se utiliza para asignar un valor a una variable:

// El operador "=" asigna valores

var numero1 = 5;

resultado = numero1 = 3; // numero1 = 3 y resultado = 3

// El operador "==" compara variables

var numero1 = 5;

resultado = numero1 == 3; // numero1 = 5 y resultado = false

Los operadores relacionales también se pueden utilizar con variables de tipo cadena de texto:

var texto1 = "hola";

var texto2 = "hola";

var texto3 = "adios";

resultado = texto1 == texto3; // resultado = false

resultado = texto1 != texto2; // resultado = false

resultado = texto3 >= texto2; // resultado = false

Cuando se utilizan cadenas de texto, los operadores "mayor que" (>) y "menor que" (<) siguen un razonamiento no intuitivo: se compara letra a letra comenzando desde la izquierda hasta que se encuentre una diferencia entre las dos cadenas de texto. Para determinar si una letra es mayor o menor que otra, las mayúsculas se consideran menores que las minúsculas y las primeras letras del alfabeto son menores que las últimas (a es menor que b, b es menor que c, A es menor que a, etc.)

**Ejercicio:**

A partir del siguiente array que se proporciona: var valores = [true, 5, false, "hola", "adios", 2];

1. Determinar cual de los dos elementos de texto es mayor
2. Utilizando exclusivamente los dos valores booleanos del array, determinar los operadores necesarios para obtener un resultado true y otro resultado false
3. Determinar el resultado de las cinco operaciones matemáticas realizadas con los dos elementos numéricos

# Estructuras de control de flujo

# Estructura if

La estructura más utilizada en JavaScript y en la mayoría de lenguajes de programación es la estructura if. Se emplea para tomar decisiones en función de una condición. Su definición formal es:

if(condicion) {

...

}

Si la condición se cumple (es decir, si su valor es true) se ejecutan todas las instrucciones que se encuentran dentro de {...}. Si la condición no se cumple (es decir, si su valor es false) no se ejecuta ninguna instrucción contenida en {...} y el programa continúa ejecutando el resto de instrucciones del script.

Ejemplo:

var mostrarMensaje = true;

if(mostrarMensaje) {

alert("Hola Mundo");

}

En el ejemplo anterior, el mensaje sí que se muestra al usuario ya que la variable mostrarMensajetiene un valor de true y por tanto, el programa entra dentro del bloque de instrucciones del if.

El ejemplo se podría reescribir también como:

var mostrarMensaje = true;

if(mostrarMensaje == true) {

alert("Hola Mundo");

}

En este caso, la condición es una comparación entre el valor de la variable mostrarMensaje y el valortrue. Como los dos valores coinciden, la igualdad se cumple y por tanto la condición es cierta, su valor es true y se ejecutan las instrucciones contenidas en ese bloque del if.

La comparación del ejemplo anterior suele ser el origen de muchos errores de programación, al confundir los operadores == y =. Las comparaciones siempre se realizan con el operador ==, ya que el operador = solamente asigna valores:

var mostrarMensaje = true;

// Se comparan los dos valores

if(mostrarMensaje == false) {

...

}

// Error - Se asigna el valor "false" a la variable

if(mostrarMensaje = false) {

...

}

La condición que controla el if() puede combinar los diferentes operadores lógicos y relacionales mostrados anteriormente:

var mostrado = false;

if(!mostrado) {

alert("Es la primera vez que se muestra el mensaje");

}

Los operadores AND y OR permiten encadenar varias condiciones simples para construir condiciones complejas:

var mostrado = false;

var usuarioPermiteMensajes = true;

if(!mostrado && usuarioPermiteMensajes) {

alert("Es la primera vez que se muestra el mensaje");

}

# Ejercicio

Completar las condiciones de los if del siguiente script para que los mensajes de los alert() se muestren siempre de forma correcta:

var numero1 = 5;

var numero2 = 8;

if(...) {

alert("numero1 no es mayor que numero2");

}

if(...) {

alert("numero2 es positivo");

}

if(...) {

alert("numero1 es negativo o distinto de cero");

}

if(...) {

alert("Incrementar en 1 unidad el valor de numero1 no lo hace mayor o igual que numero2");

}

### Estructura if...else

if(condicion) {

...

}

else {

...

}

Si la condición se cumple (es decir, si su valor es true) se ejecutan todas las instrucciones que se encuentran dentro del if(). Si la condición no se cumple (es decir, si su valor es false) se ejecutan todas las instrucciones contenidas en else { }. Ejemplo:

var edad = 18;

if(edad >= 18) {

alert("Eres mayor de edad");

}

else {

alert("Todavía eres menor de edad");

}

Si el valor de la variable edad es mayor o igual que el valor numérico 18, la condición del if() se cumple y por tanto, se ejecutan sus instrucciones y se muestra el mensaje "Eres mayor de edad". Sin embargo, cuando el valor de la variable edad no es igual o mayor que 18, la condición del if() no se cumple, por lo que automáticamente se ejecutan todas las instrucciones del bloque else { }. En este caso, se mostraría el mensaje "Todavía eres menor de edad".

El siguiente ejemplo compara variables de tipo cadena de texto:

var nombre = "";

if(nombre == "") {

alert("Aún no nos has dicho tu nombre");

}

else {

alert("Hemos guardado tu nombre");

}

La condición del if() anterior se construye mediante el operador ==, que es el que se emplea para comparar dos valores (no confundir con el operador = que se utiliza para asignar valores). En el ejemplo anterior, si la cadena de texto almacenada en la variable nombre es vacía (es decir, es igual a"") se muestra el mensaje definido en el if(). En otro caso, se muestra el mensaje definido en el bloque else { }.

La estructura if...else se puede encadenar para realizar varias comprobaciones seguidas:

if(edad < 12) {

alert("Todavía eres muy pequeño");

}

else if(edad < 19) {

alert("Eres un adolescente");

}

else if(edad < 35) {

alert("Aun sigues siendo joven");

}

else {

alert("Piensa en cuidarte un poco más");

}

**Ejercicio**

El cálculo de la letra del Documento Nacional de Identidad (DNI) es un proceso matemático sencillo que se basa en obtener el resto de la división entera del número de DNI y el número 23. A partir del resto de la división, se obtiene la letra seleccionándola dentro de un array de letras.

El array de letras es:

var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X', 'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E', 'T'];

Por tanto si el resto de la división es 0, la letra del DNI es la T y si el resto es 3 la letra es la A. Con estos datos, elaborar un pequeño script que:

1. Almacene en una variable el número de DNI indicado por el usuario y en otra variable la letra del DNI que se ha indicado. (Pista: si se quiere pedir directamente al usuario que indique su número y su letra, se puede utilizar la función prompt())
2. En primer lugar (y en una sola instrucción) se debe comprobar si el número es menor que 0 o mayor que 99999999. Si ese es el caso, se muestra un mensaje al usuario indicando que el número proporcionado no es válido y el programa no muestra más mensajes.
3. Si el número es válido, se calcula la letra que le corresponde según el método explicado anteriormente.
4. Una vez calculada la letra, se debe comparar con la letra indicada por el usuario. Si no coinciden, se muestra un mensaje al usuario diciéndole que la letra que ha indicado no es correcta. En otro caso, se muestra un mensaje indicando que el número y la letra de DNI son correctos.