FUNCIONES

Sintaxis

Una función es un conjunto de instrucciones que se agrupan para realizar una tarea concreta y que se pueden reutilizar fácilmente.

Las funciones en JavaScript se definen mediante la palabra reservada function, seguida del nombre de la función y dos paréntesis, dentro puede llevar parámetros, y su código se encuentra limitado por llaves:

function nombre_funcion(parametro1, parametro2,…) {

…

}

Dentro tiene una clausula return, que devuelve el resultado de la función. Una función sólo puede devolver un valor.

Ejemplo:

```
function producto(p1,p2) {
  return p1 * p2; // Devuelve el resultado de multiplicar p1 por p2
}
```

Llamada a una función

El código de una función se ejecuta una vez que se llama a esa función. También se suele utilizar la expresión "invocar a la función".

La llamada a la función se realiza simplemente indicando su nombre, incluyendo los paréntesis y los valores con los que se va a ejecutar, puede que ninguno.

La función termina cuando alcanza la instrucción return, devolviendo dicho valor.

Ejemplo:

```
var x = producto(4, 3);   // La función es llamada, después de la
llamada x tiene el valor 12.

function producto(a, b) {
  return a * b; }
```

- El número de argumentos que se pasa a una función debería ser el mismo que el número de argumentos que ha indicado la función. No obstante, JavaScript

no muestra ningún error si se pasan más o menos argumentos de los necesarios.

- El orden de los argumentos es fundamental, ya que el primer dato que se indica en la llamada, será el primer valor que espera la función; el segundo valor indicado en la llamada, es el segundo valor que espera la función y así sucesivamente.
- Se puede utilizar un número ilimitado de argumentos, aunque si su número es muy grande, se complica en exceso la llamada a la función.
- No es obligatorio que coincida el nombre de los argumentos que utiliza la función y el nombre de los argumentos que se le pasan.
  Ejemplo:

```
var m=7,n=8;
var x = producto(m, n);   // Después de la llamada x tiene el
valor 56.

function producto(a, b) {
  return a * b; }
```

- Si no recogemos el retorno de una función en la llamada, este valor se pierde.

Variables en las funciones

El ámbito de una variable (llamado"scope" en inglés) es la zona del programa en la que se define la variable. JavaScript define dos ámbitos para las variables: global y local.

Las variables que nos declaremos dentro de las funciones, son **variables locales**. Es recomendable ponerlas por delante la palabra reservada let. Estas variables sólo pueden ser usadas dentro de la función, al terminar la función se destruyen.

Ejemplo:

```
function suma(a, b) {
  let c;
  c=a+b;
  return c; }

suma(3,4);
document.write(c);  // Error, ya que c no es reconocida fuera de la función

// La forma correcta de obtener este valor sería:

resultado = suma(3,4);

document.write(resultado);
```

Además de variables locales, también existe el concepto de variable global, que son las que están definidas en cualquier punto del programa (incluso dentro de cualquier función).

Ejemplo:

```
var mensaje = “Mensaje de prueba”;
function muestraMensaje() {
alert(mensaje); // Muestra  -- Mensaje de prueba
}
```

Si una variable se declara fuera de cualquier función, automáticamente se transforma en variable global independientemente de si se define utilizando la palabra reservada var o no. Sin embargo, las variables definidas dentro de una función pueden ser globales o locales.

Si en el interior de una función, las variables se declaran mediante var o let, se consideran locales y las variables que no se han declarado, se transforman automáticamente en variables globales.

```
function creaMensaje() {
mensaje = "Mensaje de prueba";
}
creaMensaje();
alert(mensaje); // Muestra “Mensaje de prueba”
```

No es una buena práctica!! Lo mejor es declarar las variables globales fuera de las funciones, y locales dentro.

La recomendación general es definir como variables locales todas las variables que sean de uso exclusivo para realizar las tareas encargadas a cada función. Las variables globales se utilizan para compartir variables entre funciones de forma sencilla.