

Funciones

Cuando se desarrolla una aplicación compleja, es muy habitual utilizar una y otra vez las mismas instrucciones. Un script para una tienda de comercio electrónico por ejemplo, tiene que calcular el precio total de los productos varias veces, para añadir los impuestos y los gastos de envío.

Cuando una serie de instrucciones se repiten una y otra vez, se complica demasiado el código fuente de la aplicación, ya que:

- El código de la aplicación es mucho más largo porque muchas instrucciones están repetidas.
- Si se quiere modificar alguna de las instrucciones repetidas, se deben hacer tantas modificaciones como veces se haya escrito esa instrucción, lo que se convierte en un trabajo muy pesado y muy propenso a cometer errores.

Las funciones son la solución a todos estos problemas, tanto en JavaScript como en el resto de lenguajes de programación. Una función es un conjunto de instrucciones que se agrupan para realizar una tarea concreta y que se pueden reutilizar fácilmente.

En primer lugar se debe crear la función básica con las instrucciones comunes. Las funciones en JavaScript se definen mediante la palabra reservada *function*, seguida del nombre de la función. Su definición formal es la siguiente.

```
function nombre_funcion() {  
    ...  
}
```

La llamada a la función se realiza simplemente indicando su nombre, incluyendo los paréntesis del final y el carácter `;` para terminar la instrucción.

```
function suma_y_muestra() {  
    resultado = numero1 + numero2;  
    alert("El resultado es " + resultado);  
}  
  
var resultado;  
  
var numero1 = 3;  
var numero2 = 5;  
  
suma_y_muestra();  
  
numero1 = 10;  
numero2 = 7;  
  
suma_y_muestra();
```

Argumentos y valores de retorno

Las funciones más sencillas no necesitan ninguna información para producir sus resultados. Sin embargo, la mayoría de funciones de las aplicaciones reales deben acceder al valor de algunas variables para producir sus resultados.

Las variables que necesitan las funciones se llaman *argumentos*. Antes de que pueda utilizarlos, la función debe indicar cuántos argumentos necesita y cuál es el nombre de cada argumento. Además, al invocar la función, se deben incluir los valores que se le van a pasar a la función. Los argumentos se indican dentro de los paréntesis que van detrás del nombre de la función y se separan con una coma (,).

```
// Definición de la función
function suma_y_muestra(primerNumero, segundoNumero) { ... }
    var resultado = primerNumero + segundoNumero;
    alert("El resultado es " + resultado);
}

// Declaración de las variables
var numero1 = 3;
var numero2 = 5;

// Llamada a la función
suma_y_muestra(numero1, numero2);
```

En el código anterior, se debe tener en cuenta que:

- 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. En el ejemplo anterior, los argumentos que se pasan son `numero1` y `numero2` y los argumentos que utiliza la función son `primerNumero` y `segundoNumero`.

Afortunadamente, las funciones no solamente puede recibir variables y datos, sino que también pueden devolver los valores que han calculado. Para devolver valores dentro de una función, se utiliza la palabra reservada `return`. Aunque las funciones pueden devolver valores de cualquier tipo, solamente pueden devolver un valor cada vez que se ejecutan.

```
function calculaPrecioTotal(precio) {  
    var impuestos = 1.16;  
    var gastosEnvio = 10;  
    var precioTotal = ( precio * impuestos ) + gastosEnvio;  
    return precioTotal;  
}  
  
var precioTotal = calculaPrecioTotal(23.34);  
  
// Seguir trabajando con la variable "precioTotal"
```

Para que la función devuelva un valor, solamente es necesario escribir la palabra reservada `return` junto con el nombre de la variable que se quiere devolver. En el ejemplo anterior, la ejecución de la función llega a la instrucción `return precioTotal;` y en ese momento, devuelve el valor que contenga la variable `precioTotal`.

Si no se indica el nombre de ninguna variable, JavaScript no muestra ningún error y el valor devuelto por la función simplemente se pierde y por tanto, no se utilizará en el resto del programa. En este caso, tampoco es obligatorio que el nombre de la variable devuelta por la función coincida con el nombre de la variable en la que se va a almacenar ese valor.

Si la función llega a una instrucción de tipo `return`, se devuelve el valor indicado y finaliza la ejecución de la función. Por tanto, todas las instrucciones que se incluyen después de un `return` se ignoran y por ese motivo la instrucción `return` suele ser la última de la mayoría de funciones.

Para que el ejemplo anterior sea más completo, se puede añadir otro argumento a la función que indique el porcentaje de impuestos que se debe añadir al precio del producto. Evidentemente, el nuevo argumento se debe añadir tanto a la definición de la función como a su llamada

```
function calculaPrecioTotal(precio, porcentajeImpuestos) {  
    var gastosEnvio = 10;  
    var precioConImpuestos = (1 + porcentajeImpuestos/100) * precio;  
    var precioTotal = precioConImpuestos + gastosEnvio;  
    return precioTotal.toFixed(2);  
}  
  
var precioTotal = calculaPrecioTotal(23.34, 16);
```