Lenguaje de Marcas: Javascript

Iván Nieto Ruiz

il.nietoruiz@edu.gva.es

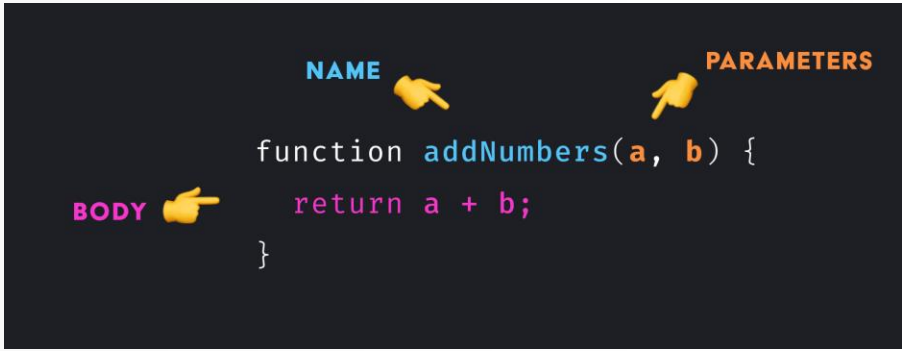
¿Qué es una función?

Una **función** es un bloque de código reutilizable diseñado para realizar una tarea específica.

Las funciones nos ayudan a **organizar, reutilizar y simplificar el código**.

Ventajas:

- Evita repetir código.
- Facilita el mantenimiento.
- Mejora la legibilidad.




```
function saludar() {
	console.log("Esta es una función básica en JavaScript");
}

//para ejecutarla simplemente la invocamos por su nombre:
saludar();
```

Funciones con Parámetros y Argumentos

Podemos **pasar valores a una función** a través de **parámetros**. Esto nos permite trabajar con datos dinámicos.

```
function saludar(nombre) {
    console.log("Hola, " + nombre + "!");
}

saludar("Carlos"); // Salida: Hola, Carlos!
saludar("Ana"); // Salida: Hola, Ana!
```

Funciones que Devuelven Valores

```
function sumar(a, b) {
    return a + b;
}

let resultado = sumar(5, 3);

console.log("Resultado:", resultado);
```

Funciones Expresadas (Funciones Anónimas)

Una **función expresada** es aquella que se almacena en una variable. A menudo, estas funciones no tienen nombre y se denominan **funciones anónimas**.

```
const multiplicar = function(a, b) {
      return a * b;
};

console.log(multiplicar(4, 5));
```

La función **no tiene nombre** dentro de function(...) { ... }.
Se almacena en la variable sumar, que ahora **funciona como el nombre de la función**.
Se llama usando sumar(3, 5), como si fuera una función con nombre.

Ventajas de las funciones anónimas

- Se pueden asignar a variables.
- Son útiles para operaciones **cortas y específicas**.

Arrow Functions (Funciones de Flecha)

Una **función expresada** es aquella que se almacena en una variable. A menudo, estas funciones no tienen nombre y se denominan **funciones anónimas**.

```
const multiplicar = (a, b) => a * b;

console.log(multiplicar(3, 5));
```

- **Se eliminan `{}` y `return`** porque la función **solo tiene una línea**.
- La flecha => indica el retorno automático del resultado.

Ventajas de las funciones anónimas

- Se pueden asignar a variables.
- Son útiles para operaciones **cortas y específicas**.
- Si solo tenemos un argumento, podemos obviar el paréntesis:

Funciones

Función anónima normal

```
const cuadrado = function(num) {
return num * num;
};
console.log(cuadrado(4));
```

Función Arrow

```
const cuadrado = num => num * num;
console.log(cuadrado(4));
```

Importante:

- **Funciones anónimas** son útiles en **callbacks** y cuando **no necesitas reutilizarlas**.
- **Funciones de flecha** son una alternativa más **concisa y moderna**.
- **Las funciones de flecha no tienen this**, por lo que no siempre son ideales.
- **Usa funciones normales** cuando necesites acceder a this en objetos o eventos.

¿Preguntas?