Funciones y Alcance en JavaScript

Julian F. Latorre

15 de octubre de 2024

Objetivos de la Sesión

- Comprender qué son las funciones
- Aprender a declarar y llamar funciones
- Entender los diferentes tipos de funciones en JavaScript
- Explorar el concepto de alcance (scope) y su importancia
- Practicar con ejemplos y ejercicios interactivos

¿Qué son las funciones?

- Bloques de construcción fundamentales en JavaScript
- "Subprogramas" que realizan tareas específicas
- Permiten:
  - Organizar y estructurar el código
  - Reutilizar código
  - Mejorar la legibilidad y mantenibilidad
  - Abstraer operaciones complejas

Sintaxis básica de una función

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

saludar();  // Imprime: Hola , mundo!
```

- `function`: palabra clave
- `saludar`: nombre de la función
- `()`: pueden contener parámetros
- `{}`: cuerpo de la función

Funciones con parámetros

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

saludarPersona("María");   //  Hola , María!
saludarPersona("Juan");    //  Hola , Juan!
```

- `nombre`: parámetro
- "María", "Juan": argumentos

Funciones con retorno

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

let resultado = sumar(5, 3);
console.log(resultado);  // Imprime: 8
```

- `return`: devuelve un valor
- La función termina cuando se ejecuta `return`

Ejercicio práctico

- Escriban una función llamada `calcularArea`
- Debe tomar el radio de un círculo como parámetro
- Debe devolver el área del círculo
- Recuerden: el área de un círculo es πr^2

Tipos de Funciones: Declarativas vs. Expresiones

```
// Función declarativa
function saludar() {
  console.log("Hola");
}

// Expresión de función
const despedir = function() {
  console.log("Adiós");
};
```

- Las funciones declarativas se "elevan" (hoisting)
- Las expresiones de función deben definirse antes de usarse

Funciones Flecha (Arrow Functions)

```
// Función tradicional
const sumar = function(a, b) {
  return a + b;
};

// Función flecha equivalente
const sumarFlecha = (a, b) => a + b;
```

- Sintaxis más concisa
- Útiles para funciones cortas y como argumentos

Ejercicio práctico

- Tomen la función `calcularArea` que escribieron antes
- Conviértanla en una función flecha

Alcance (Scope) en JavaScript

- Se refiere a la visibilidad de las variables
- Determina qué partes del código tienen acceso a una variable
- Tipos principales:
  - Alcance global
  - Alcance local (de función)
  - Alcance de bloque (con `let` y `const`)

Alcance Global vs. Local

```
let globalVar = "Soy global";

function ejemploScope() {
  let localVar = "Soy local";
  console.log(globalVar);  // Accesible
  console.log(localVar);   // Accesible
}

ejemploScope();
console.log(globalVar);  // Accesible
console.log(localVar);   // Error
```

Alcance de Bloque con let y const

```
if (true) {
  var x = 10;
  let y = 20;
  const z = 30;
}

console.log(x);  // Imprime: 10
console.log(y);  // Error: y is not defined
console.log(z);  // Error: z is not defined
```

- `var`: alcance de función o global
- `let` y `const`: alcance de bloque

Ejercicio final

- Escriban una función que demuestre:
  - Uso de variables globales
  - Uso de variables locales
  - Uso de variables de bloque
- La función debe intentar acceder a estas variables en diferentes puntos
- Registren en consola si el acceso fue exitoso o no

Conclusión y Resumen

- Definición y llamada de funciones
- Funciones con parámetros y valor de retorno
- Diferentes tipos de funciones:
  - Declarativas
  - Expresiones
  - Flecha
- Alcance: global, local y de bloque

¿Preguntas?