Apuntes: Funciones en JavaScript

Introducción: Las funciones son bloques de código reutilizables que realizan tareas específicas. Mejoran la modularidad y organización del código. Existen varias maneras de declararlas.

I. Declaración de Funciones:

- Sintaxis tradicional: `function nombreFuncion(parámetros){ cuerpo de la función }`
- Ejemplo:

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

II. Expresiones de Función:

- Se asignan a una variable. Útiles para funciones anónimas o asignación dinámica.
- Ejemplo:

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

III. Funciones Flecha (Arrow Functions):

- Sintaxis más concisa (ES6). Ideales para funciones cortas y anónimas.
- Omisión de `return` y llaves si solo hay una expresión.
- Ejemplo:

```
const sumar = (a, b) => a + b;
console.log(sumar(2, 3)); // 5

const saludar = (nombre) => console.log("Hola, " + nombre + "!");
saludar("Juan"); // Hola, Juan!
```

IV. Parámetros:

- Valores de entrada para la función.
- Parámetros por defecto para valores opcionales.
- Operador spread (`...`) para un número variable de argumentos (ES6).
- Ejemplo:

```
function saludar(nombre = "Invitado") {
```

```
  console.log("Hola, " + nombre + "!");
}
saludar(); // Hola, Invitado!
saludar("Ana"); // Hola, Ana!

function sumar(...numeros) {
  return numeros.reduce((total, num) => total + num, 0);
}
console.log(sumar(1, 2, 3, 4)); // 10
```

V. Retorno de Valores:

- `return` especifica el valor devuelto (o `undefined` si no se usa).
- Puede retornar cualquier tipo de dato.
- Ejemplo:

```
function crearObjeto(nombre, edad) {
  return { nombre: nombre, edad: edad };
}
const persona = crearObjeto("Pedro", 30);
console.log(persona); // {nombre: "Pedro", edad: 30}
```

VI. Ámbito (Scope):

- Variables declaradas dentro de una función tienen ámbito local (solo accesibles dentro).
- Variables fuera de la función tienen ámbito global.
- Ejemplo:

```
let x = 10;
function miFuncion() {
  let x = 5;
  console.log(x); // 5
}
miFuncion();
console.log(x); // 10
```

VII. Funciones como Objetos de Primera Clase:

- Pueden ser pasadas como argumentos, devueltas por otras funciones, y asignadas a variables.
- Ejemplo:

```
function aplicarOperacion(a, b, operacion) {
  return operacion(a, b);
}
const sumar = (x, y) => x + y;
const restar = (x, y) => x - y;
console.log(aplicarOperacion(5, 3, sumar)); // 8
console.log(aplicarOperacion(5, 3, restar)); // 2
```

VIII. Métodos de Función (`call()`, `apply()`, `bind()`):

- Controlan el contexto (`this`) y la forma de pasar argumentos.
- Ejemplo:

```
const persona = {
  nombre: "Maria",
  saludar: function () {
    console.log("Hola, mi nombre es " + this.nombre);
  },
};
persona.saludar(); // Hola, mi nombre es Maria
const saludar = persona.saludar.bind({ nombre: "Ana" });
saludar(); // Hola, mi nombre es Ana
```