**Introducción a estructuras de datos**

Las **estructuras de datos** son una forma de organizar y almacenar datos en una computadora de manera eficiente para que puedan ser utilizados y manipulados de forma más efectiva. En JavaScript, dos de las estructuras de datos más comunes y esenciales que usaremos son arreglos (**arrays**) y objetos (**objects**). Estos son fundamentales para la programación en JavaScript, ya que permiten almacenar colecciones de datos y manipularlos de manera flexible y eficiente.

A lo largo de esta lectura, veremos qué son los arreglos y objetos, cómo se usan y qué aplicaciones prácticas tienen en el desarrollo de software.

**Arreglos (Arrays)**

Un **arreglo** es una estructura de datos que almacena una colección de elementos. Los arreglos en JavaScript pueden contener cualquier tipo de dato: números, cadenas de texto, objetos e incluso otros arreglos. Los elementos en un arreglo se organizan en una secuencia y se accede a ellos mediante un índice, el cual comienza en 0.

**Cómo crear un arreglo**

Para crear un arreglo en JavaScript, usamos corchetes []:

let frutas = ["manzana", "plátano", "cereza"];

console.log(frutas[0]);  *// Imprime "manzana"*

**Operaciones comunes con arreglos**

**Acceder a un elemento**: Puedes acceder a un elemento del arreglo usando su índice.

let colores = ["rojo", "verde", "azul"];

console.log(colores[1]);  *// Imprime "verde"*

**Agregar elementos**: Usamos el método .push() para agregar elementos al final del arreglo.

colores.push("amarillo");

console.log(colores);  // Imprime ["rojo", "verde", "azul", "amarillo"]

**Eliminar elementos**: Usam os el método .pop() para eliminar el último elemento del arreglo.

colores.pop();

console.log(colores);  // Imprime ["rojo", "verde", "azul"]

**Ejemplo de caso de uso de arreglos**

Imagina que estás organizando una fiesta y necesitas llevar un registro de los invitados. Usar un arreglo es una manera sencilla de almacenar y manejar una lista de nombres.

let invitados = ["Juan", "María", "Pedro", "Ana"];

invitados.push("Luis");  // Agrega a Luis a la lista

console.log(invitados);  // Imprime ["Juan", "María", "Pedro", "Ana", "Luis"]

**Objetos (Objects)**

Un **objeto** en JavaScript es una estructura que permite almacenar una colección de datos en forma de propiedades clave-valor. A diferencia de los arreglos, los objetos no están ordenados por índice, sino por claves (o propiedades). Las claves son siempre cadenas de texto, y los valores pueden ser cualquier tipo de dato (números, cadenas, arreglos, funciones, etc.).

**Cómo crear un objeto**

Para crear un objeto en JavaScript, usamos llaves {}:

let persona = {

    nombre: "Juan",

    edad: 30,

    ciudad: "Madrid"

};

console.log(persona.nombre);  *// Imprime "Juan"*

**Operaciones comunes con objetos**

**Acceder a una propiedad**: Puedes acceder a una propiedad del objeto usando el nombre de la propiedad.

console.log(persona.edad); *// Imprime 30*

**Modificar una propiedad**: Puedes cambiar el valor de una propiedad asignando un nuevo valor.

persona.ciudad = "Barcelona"*;*

console.log(persona.ciudad)*;  // Imprime "Barcelona"*

**Agregar una nueva propiedad**: Puedes agregar propiedades nuevas a un objeto de manera dinámica.

persona.telefono = "123-456-7890"*;*

console.log(persona.telefono)*;  // Imprime "123-456-7890"*

**Ejemplo de caso de uso de objetos**

Un objeto es útil cuando deseas almacenar información relacionada entre sí, como los detalles de una persona. Por ejemplo, si quieres almacenar los datos de un libro, podrías hacerlo usando un objeto con propiedades como titulo, autor y año.

let libro = {

    titulo: "Cien años de soledad",

    autor: "Gabriel García Márquez",

    año: 1967

};

console.log(libro.titulo);  *// Imprime "Cien años de soledad"*

**Comparación entre Arreglos y Objetos**

Ambos, **arreglos** y **objetos**, son fundamentales en JavaScript, pero tienen usos diferentes:

* Los arreglos son ideales cuando necesitas almacenar una colección ordenada de elementos, y acceder a ellos por un índice numérico.
* Los objetos son mejores cuando necesitas almacenar una colección no ordenada de datos relacionados entre sí, donde cada elemento tiene un nombre o clave asociada.

**Ejemplo de uso combinado**

Supongamos que deseas llevar un registro de varios productos en una tienda. Puedes usar un arreglo para almacenar los productos y un objeto para describir cada uno de esos productos.

let productos = [

    { nombre: "Camiseta", precio: 20, stock: 50 },

    { nombre: "Pantalón", precio: 40, stock: 30 },

    { nombre: "Zapatos", precio: 60, stock: 20 }

];

console.log(productos[1].nombre);  // Imprime "Pantalón"

En este ejemplo, el arreglo productos almacena los objetos que contienen la información de cada producto. Cada objeto tiene claves como nombre, precio y stock.

**Casos de Uso en el Mundo Real**

**- Arreglos**:

1. Lista de tareas: Usar un arreglo para almacenar las tareas que necesitas realizar en un día.
2. Carrito de compras: Un arreglo puede almacenar los productos que el usuario añade a su carrito.
3. Orden de empleados: Puedes usar un arreglo para almacenar los nombres de los empleados en un proyecto, asegurándote de que se sigan ciertos pasos o etapas.

- **Objetos**:

1. Registro de usuario: Un objeto puede almacenar información sobre un usuario (nombre, dirección, email).
2. Configuración de una aplicación: Los objetos se usan para almacenar configuraciones o parámetros de una aplicación, como el tema de la interfaz de usuario, el idioma, etc.
3. Receta de cocina: Un objeto puede almacenar los ingredientes y las instrucciones de una receta, cada uno con su nombre y cantidad.

**Conclusión**

En esta lectura, hemos cubierto dos de las estructuras de datos más importantes en JavaScript: **arreglos** y **objetos**.

- **Arreglos**: Son perfectos para almacenar colecciones ordenadas de datos que se pueden acceder mediante índices. Son útiles cuando el orden de los elementos es importante.

- **Objetos**: Son ideales para almacenar datos relacionados entre sí bajo una estructura de clave-valor. Son más flexibles que los arreglos, ya que puedes usar cualquier tipo de clave.

Ambos tipos de estructuras son fundamentales para manejar datos en JavaScript y permiten a los programadores organizar y manipular información de manera eficiente. Aprender a usarlos correctamente es esencial para desarrollar programas más complejos y funcionales. ¡Ahora que conoces las bases, puedes empezar a aplicar estas estructuras en tus propios proyectos y tareas!