# Capítulo 06: Conceptos de JavaScript necesarios para trabajar con Next.js

En este capítulo presentamos los conceptos clave de JavaScript que los alumnos deben dominar para poder trabajar con Next.js (y React dentro de Next.js). Cada concepto incluye ejemplos prácticos y se señala su aplicación en FrontEnd y BackEnd.

## 1. Variables (let, const)

let: se puede reasignar.  
const: no se puede reasignar.  
  
Ejemplo:  
let contador = 1;  
contador = 2; // ✅ permitido  
  
const PI = 3.14;  
// PI = 3.15 ❌ error  
  
📌 FrontEnd: estados suelen declararse con const.  
📌 BackEnd: configuración de puertos, constantes globales.

## 2. Funciones y Arrow Functions

Ejemplo:  
// Función normal  
function sumar(a, b) { return a + b; }  
  
// Arrow function  
const restar = (a, b) => a - b;  
  
📌 FrontEnd: eventos en React.  
📌 BackEnd: callbacks en Express.

## 3. Condicionales y Operadores

const edad = 20;  
if (edad >= 18) { console.log("Es mayor de edad"); }  
  
const tienePermiso = true;  
if (edad >= 18 && tienePermiso) { console.log("Puede ingresar"); }  
  
📌 FrontEnd: mostrar botones o secciones condicionales.  
📌 BackEnd: validación de permisos.

## 4. Bucles y Métodos de Arrays

const numeros = [1,2,3];  
  
for (let i=0; i<numeros.length; i++) console.log(numeros[i]);  
  
numeros.forEach(n => console.log(n));  
  
const cuadrados = numeros.map(n => n\*n); // [1,4,9]  
  
📌 FrontEnd: renderizar listas en React.  
📌 BackEnd: procesar registros de una base de datos.

## 5. Destructuring

De objetos:  
const persona = { nombre: "Ana", edad: 25 };  
const { nombre, edad } = persona;  
  
Con alias:  
const { nombre: fullName } = persona;  
  
Con valores por defecto:  
const { edad = 18 } = persona;  
  
Anidado:  
const { perfil: { ciudad } } = usuario;  
  
De arrays:  
const [a,b] = [10,20];  
  
Ignorando elementos:  
const [,segundo] = [10,20,30];  
  
📌 FrontEnd: const { title } = props.  
📌 BackEnd: const { user, pass } = req.body.

## 6. Spread Operator (... expandir)

Copiar arrays:  
const arr2 = [...arr1];  
  
Combinar arrays:  
const combinado = [...a, ...b];  
  
Copiar objetos:  
const copia = { ...user };  
  
Actualizar objetos:  
const actualizado = { ...user, edad: 26 };  
  
Pasar parámetros:  
Math.max(...nums);  
  
📌 FrontEnd: setItems([...items, newItem]).  
📌 BackEnd: clonar objetos antes de guardarlos.

## 7. Rest Operator (... agrupar)

Capturar parámetros variables:  
function sumar(...nums) { return nums.reduce((a,b)=>a+b,0); }  
  
Desestructuración con resto:  
const [primero, ...resto] = [1,2,3];  
  
En objetos:  
const { id, ...datos } = usuario;  
  
📌 FrontEnd: props dinámicos.  
📌 BackEnd: separar campos en payloads.

## 8. Objetos Literales

const producto = { id: 1, nombre: "Laptop", precio: 1500 };  
  
📌 FrontEnd: props.  
📌 BackEnd: JSON en APIs.

## 9. Módulos (Imports/Exports)

// operaciones.js  
export const sumar = (a,b)=>a+b;  
  
// main.js  
import { sumar } from "./operaciones.js";  
  
📌 FrontEnd: importar componentes.  
📌 BackEnd: importar controladores.

## 10. Asincronía (Promesas / async-await)

fetch("https://api.com").then(res=>res.json()).then(data=>console.log(data));  
  
async function cargar(){ const res = await fetch("https://api.com"); const data = await res.json(); }  
  
📌 FrontEnd: llamadas a APIs.  
📌 BackEnd: consultas a base de datos.

## 11. DOM vs Virtual DOM

// Vanilla JS  
const btn = document.getElementById("btn");  
btn.addEventListener("click",()=>alert("Hola"));  
  
// React (Next.js)  
<button onClick={() => alert("Hola")}>Click</button>  
  
📌 Diferencia: en React no manipulamos el DOM directamente, lo hace el Virtual DOM.

## Resumen pedagógico

- Estos conceptos son la base para comprender React dentro de Next.js.  
- En FrontEnd permiten manejar JSX, props, estado, listas y condicionales.  
- En BackEnd permiten trabajar con APIs, DB y asincronía.  
  
👉 Con ellos sólidos, el alumno puede entrar directo a Next.js y comprender cómo unir FrontEnd y BackEnd.