## Node.js - JavaScript on Server

What is Node.js?

* Node.js allows developers to use JavaScript on both
  + the client-side
  + and the server-side
* It provides a unified language and ecosystem.

What is a Server?

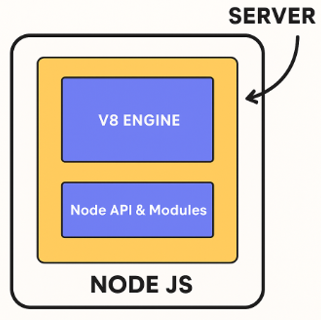
* A server is just a remote computer.
* You can think of it as:
  + A CPU running remotely.
  + A computer that provides resources and services over a network to another computer or program.

V8 Engine - (Google’s Engine)

* The JavaScript engine used in Node.js is V8.
* V8 is written in C++.
* It’s Google’s open-source, high-performance engine used in:
  + Chrome browser
  + Node.js
* It also supports WebAssembly.

V8 and Node.js

* V8 can be embedded into any C++ application.
* Node.js is a C++ application that has V8 embedded into it.



ECMA Script

* ECMA Script is the standard or set of rules that JavaScript engines follow.
* The JavaScript engine follows these standards.

What does standard or set of rules mean in ECMAScript?

Imagine a Game…

Let’s say you're playing cricket.  
For everyone to enjoy the game, we need:

* A common set of rules
* Everyone should agree on how to bowl, bat, and score

That rulebook is the standard for playing cricket.

**Now Think of JavaScript…**

JavaScript is a language that runs in browsers (like Chrome, Firefox, Safari, etc.)

Each browser has a JavaScriptengine (like:

* Chrome → V8
* Firefox → SpiderMonkey
* Safari → JavaScriptCore)

If these engines do not follow the same rules, your code might work in one browser but break in another.

**ECMAScript is the official rulebook for JavaScript.**

It defines how the language should behave.

### What Kind of Rules Does It Define?

It specifies the behavior of core language features, such as:

* What let, const, and function declarations mean.
* How strict equality (===) and type coercion (==) should behave.
* How Promises work, including methods like .then() and .catch().
* Scope handling, hoisting, async/await behavior, and much more.

### Why It Matters in Real-World Development ?

***Without ECMAScript -***

* JavaScript would behave **differently in different browsers**
* Your code might run in Chrome but break in Firefox

***Because of ECMAScript -***

* You get **consistent behavior** across all modern environments
* Tools like **TypeScript, Babel, and ESLint** rely on this spec to offer reliable features and linting

Why is V8 written in C++? What is the benefit?

* When we write JavaScript code, it is high-level code.
* V8 (written in C++) is used to:
  + Convert that high-level JS code into low-level machine code.
  + This machine code is called binary — the only form understood by computers.

### **Flow of JavaScript Code Execution**

#### **1. You Write JavaScript Code**

* A high-level, human-readable programming language
* Not directly understood by machines

#### **2**. **JavaScript Engine (e.g., V8)**

* **Parses** the code and generates an **AST** (Abstract Syntax Tree)
* **Compiles** it into **bytecode** (intermediate low-level code)
* **Optimizes** frequently executed code using **JIT (Just-In-Time) compilation**
* Produces highly optimized **machine code**

#### **3. Machine Code**

* Translated into **binary (0s and 1s)** - native CPU instructions
* This binary code is what the **computer actually executes**

What is Bytecode?

Bytecode is an intermediate code between your JavaScript source code and the final machine code.

***Think of it as -***

* A simplified, lower-level version of your code
* Not human-readable, but not yet binary
* Easier for the engine to optimize and execute
* Still needs to be converted into machine code

***Used by -***

* The Ignition interpreter in V8 engine
* Portable: bytecode can run across different hardware if the engine supports it

***Example -***

let x = 10;

The engine turns this into bytecode like:

LdaSmi [10]

Star r0

What is Machine Code?

Machine code is the final native code (in binary) that your CPU can execute directly.

***Key Points -***

* It's CPU-specific (Intel, ARM, etc.)
* Fully optimized by the engine's JITcompiler
* Made up of 0s and 1s - actual instructions the processor understands

How JavaScript Code Translates into Machine-Level Instructions ?

Write JS code

↓

JavaScript Code

↓

C++ V8 Engine (converts it)

↓

Assembly Code / Machine Code

↓

Binary Code (Understood by the computer)

