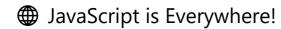
JS Engine.md 2025-05-02

Episode 16: JS Engine Exposed – Google's V8 Architecture



From **smartwatches ②**, **robots ③**, to **browsers ③** — JavaScript runs on everything thanks to the **JavaScript Runtime Environment (JRE) %**.

What is JRE (JavaScript Runtime Environment)?

- Think of it as a **container** filled with everything needed to run JS code:
 - **S IS Engine** (core of JRE)
 - **Web APIs** to interact with the outside world
 - 🖾 Event Loop
 - Z Callback Queue
 - # Microtask Queue

Browsers like Chrome St can run JS because they come with a built-in JRE.

📃 ECMAScript & JS Engines

JS Engines follow ES rules. Some popular ones:

- ♦ V8 Chrome, Edge (by Google)
- **SpiderMonkey** Firefox (created by Brendan Eich himself!)
- Chakra Older versions of Edge (now deprecated)

What is a JavaScript Engine?

How JavaScript Code Runs Inside the Engine

JS execution goes through 3 core stages:

1. Parsing

Breaking code into understandable chunks for the engine.

JS Engine.md 2025-05-02

© Example:

```
let a = 7;
```

Becomes tokens:

- let, a, =, 7
- Then, the Syntax Parser converts these tokens into an AST (Abstract Syntax Tree) &
- **%** Use **AST** Explorer to visualize it.

It's like a package.json for your line of JS code.

② 2. Compilation (JIT - Just-In-Time)

Combines benefits of interpreter 🗣 and compiler 🖹

Old days:

JS was only interpreted.

Now:

JS is **JIT compiled**:

- **S** JS compiles **during execution**!
- ✓ Yes, JavaScript does compile!
 ✓

3. Execution

The moment JS code actually runs 🏂

Needs:

- @ Memory Heap for storing variables, objects
- **Call Stack** stack of function calls (from previous episodes)
- 🔏 Garbage Collector uses Mark & Sweep 👶 to free up unused memory

% V8 Architecture (Google's JS Engine)

Google's V8 is one of the **most powerful JS engines**, powering Chrome & Node.js &

- Components:
 - **Q Ignition** Interpreter

JS Engine.md 2025-05-02

- **TurboFan** Optimizing Compiler
- **A Orinoco** Smart Garbage Collector

⊕ Flow:

```
Your Code ☐

↓

Parsing ♠

AST ♠

↓

Ignition (Interpreter) ֎ + TurboFan (Compiler) 

Bytecode / Optimized Machine Code

↓

Execution ♠
```

Summary with Emojis:

Concept	Emoji	Meaning
JS Engine	(Core software that runs JS
Parsing	•	Tokenizing + AST generation
JIT Compilation		Compile + Interpret during runtime
Execution	ô	Running the bytecode
Memory Heap	@	Stores variables, objects, etc.
Call Stack	6	Tracks function execution
Garbage Collection	S	Cleans unused memory (Mark & Sweep)
Ignition	P	Interpreter in V8
TurboFan	Ð	Optimizing compiler in V8
Orinoco	۵	Garbage collector in V8

Fun Fact:

Different companies build different JS engines to **optimize performance**, but all must follow ECMAScript 📓 standards!