

JavaScript Engine (V8, SpiderMonkey)

What is a JavaScript Engine?

A **JavaScript Engine** is a program or interpreter that **executes JavaScript code**. It takes human-written JavaScript code and translates it into **machine code** that a computer can understand and run.

JavaScript engines are embedded in web browsers (and in environments like Node.js) to interpret or compile JavaScript.

How a JavaScript Engine Works

The process typically includes:

1. **Parsing:** Reads the JavaScript code and creates an Abstract Syntax Tree (AST).
 2. **Compilation / Interpretation:**
 - Earlier engines **interpreted** JavaScript line-by-line (slower).
 - Modern engines use **Just-In-Time (JIT) compilation** to convert code into machine code at runtime for faster performance.
 3. **Execution:** The compiled machine code is executed by the CPU.
 4. **Garbage Collection:** Unused memory is cleaned up automatically to optimize performance.
-

Popular JavaScript Engines

1. V8 (by Google)

- **Used in:** Chrome, Edge, Node.js, Deno

- **Language:** Written in C++
- **Open-source:** Yes (hosted on GitHub)

Key Features:

- **JIT Compiler:** Converts JS into machine code at runtime.
- **Ignition:** The interpreter that generates bytecode.
- **TurboFan:** The optimizing compiler that converts bytecode to high-performance machine code.
- **Garbage Collector:** Automatically manages memory.
- **Supports WebAssembly:** Can run low-level code efficiently.

Flow in V8:

JavaScript → Parser → AST → Bytecode (Ignition) → Optimized Machine Code (TurboFan)

Advantages:

- Extremely fast execution
- Continuous optimization
- Powers both frontend (Chrome) and backend (Node.js)

2. SpiderMonkey (by Mozilla)

- **Used in:** Firefox
- **Language:** Written in C++
- **Open-source:** Yes

Key Features:

- **Baseline Interpreter:** Quickly converts code into bytecode.
- **IonMonkey:** The optimizing JIT compiler for performance.

- **Garbage Collection:** Uses incremental and generational GC.
- Supports **ES6+** features and WebAssembly.

Flow in SpiderMonkey:

JavaScript → Parser → AST → Bytecode → JIT Compilation (IonMonkey)

Development Use:

- Also used for embedding JavaScript into C++ applications
- Used by Mozilla in other products besides Firefox

Other JavaScript Engines

Engine	Organization	Used In	Notes
Chakra	Microsoft	Old versions of Edge	Replaced by V8 in Chromium-based Edge
JavaScriptCore	Apple	Safari	Also known as Nitro
Hermes	Meta (Facebook)	React Native (mobile apps)	Optimized for low memory footprint

Why JIT Compilation is Important

JIT (**Just-In-Time**) compilers in V8 and SpiderMonkey boost performance by:

- Compiling frequently used code paths into native machine code.
- Skipping repetitive interpretation.
- Making dynamic language execution faster (even though JS is not statically typed).

V8 vs SpiderMonkey – Quick Comparison

Feature	V8	SpiderMonkey
---------	----	--------------

Developed by	Google	Mozilla
Used in	Chrome, Node.js, Deno	Firefox
Optimizing Compiler	TurboFan	IonMonkey
Interpreter	Ignition	Baseline
GC Type	Generational	Incremental + Generational
Performance	Very high	High
WebAssembly Support	Yes	Yes
Open Source	Yes	Yes

Summary

- JavaScript engines **power modern web and backend apps** by converting JS to machine code.
- **V8** (Google) is the most widely used engine — fast, optimized, and used in Chrome and Node.js.
- **SpiderMonkey** (Mozilla) is robust and used in Firefox.
- Both use **JIT compilers** and support modern JS features and **WebAssembly**.