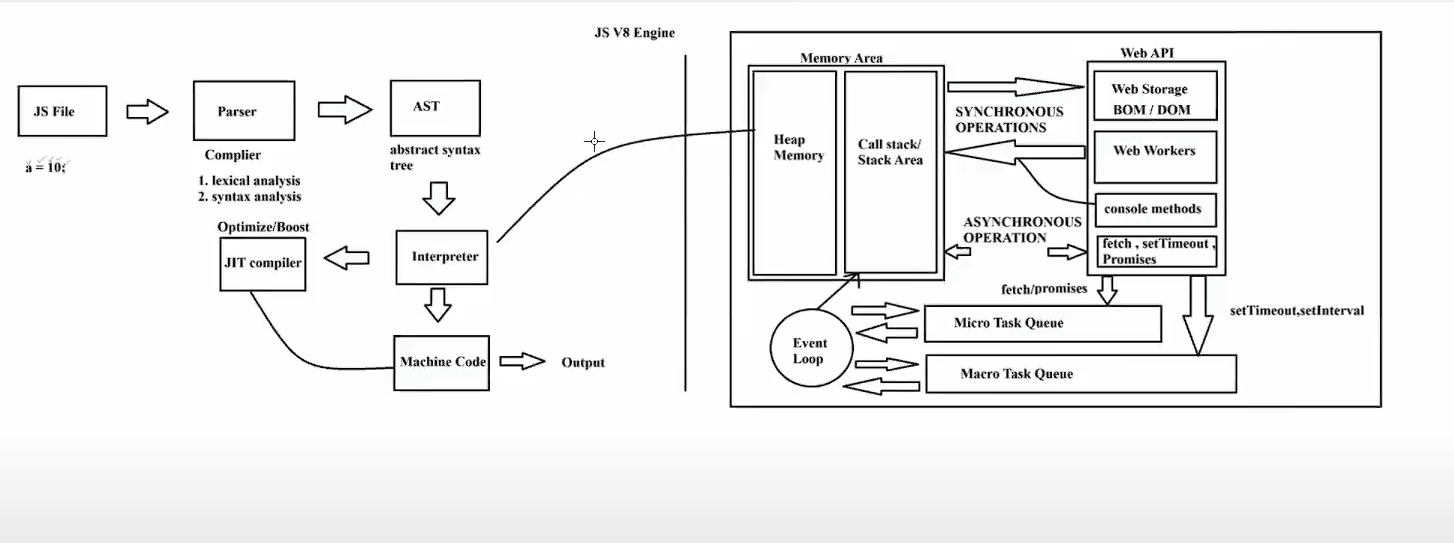
JAVASCRIPT V8 ENGINE(GOOGLE BROWSER)



The following are key Components In JavaScript V8 Engine

1. Parser
2. Abstract Syntax Tree(AST)
3. Interpreter
4. JIT Compiler
5. Memory Area
6. Web API
7. Call stack Queue
8. Event Loop

PARSER

A parser is the component of a large file system like compiler/ Interpreter that analyses the syntax of source code and gives the output in the form of syntax tree/Abstract syntax tree

Key Functions of Parser:

1. Syntax Checking
2. Token Consumption
3. Tree Generation

Output of a Parser:

1. Syntax Tree
2. Abstract Syntax Tree

ABSTRACT SYNTAX TREE

A simplified tree that omits unnecessary details like parenthesis for easier Processing by later stages

INTERPRETER

A complier is a program/software that translate the written source code into machine understandable language. The output of compiler is a machine code or Byte Code

A compiler consists of multiple phases, one of which is parsing

Key functions of a compiler

1. Lexical Analysis
2. Syntax Analysis
3. Semantic Analysis
4. Intermediate Code Generation
5. Optimization
6. Code Generation

JIT COMPILER:

**Just-In-Time (JIT) Compiler** is a part of the V8 engine that helps optimize the execution of JavaScript code by **compiling it to machine code during runtime.**  It helps Interpreter to compile the code faster.

MEMORY AREA:

A Browser engine handles the execution of javaScript code and memory management. It Primarily uses two main memory areas called Heap and Call Stack(stack Area). These both are part of JS runtime environment and Responsible for storing And executing code

1. HEAP MEOMRY

Used for storing object, variables and functions dynamically allocated during program execution

Stores large objects like arrays, objects and clousers. In this memory is allocated and freed dynamically by the Javascript engines Grabage Collector

1. CALL STACK/STACK AREA

Manages the execution context of code Including Functions calls and their local Variables. It Works on LAST IN FAST OUT(LIFO) Principle. It stores primitive data types and function call contents function

Each function call adds a new stack frame to call stake, when the function completes, Its stack frame is removed.

Call stack area is Limited Size When compared to Heap Memory

WEB API

Web APIs are **external APIs provided by the browser** (not part of the JavaScript language itself) to perform tasks asynchronously, like handling timers, making network requests, or interacting with the DOM.

JavaScript is a single-threaded language, meaning it can execute only one line of code at a time. To handle **asynchronous tasks**, the browser provides these Web APIs.

CALL STACK QUEUE

The **Call Stack** in JavaScript is a **data structure** that manages the execution of functions. It works on the **Last In, First Out (LIFO)** principle.

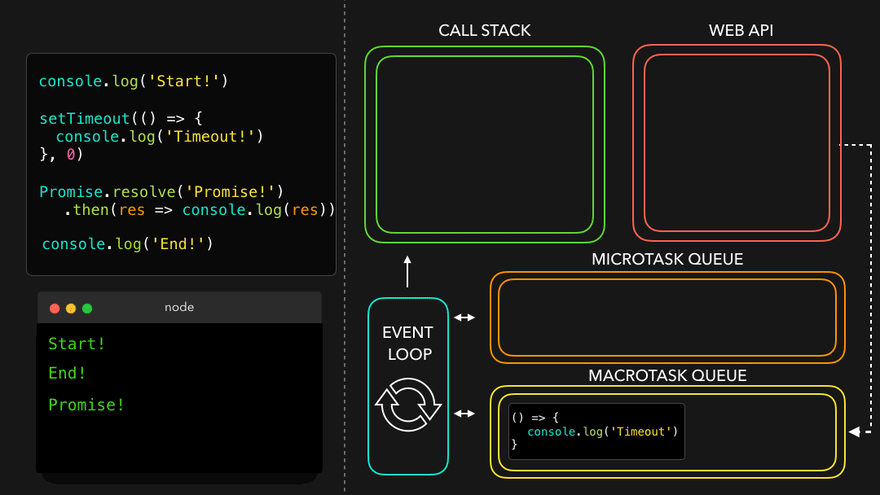
* Whenever a function is called, it gets **pushed** onto the Call Stack.
* When the function completes, it is **popped** off the stack.

EVENT LOOP

The Event Loop is a mechanism that ensures the **JavaScript Call Stack remains non-blocking**. It continuously monitors the **Call Stack** and the **Call Stack Queue**

**Key Points:**

1. The Event Loop checks if the **Call Stack is empty**.
2. If the Call Stack is empty and there are pending callbacks in the **Call Stack Queue** The callback function is moved from the Queue to the **Call Stack**.
3. This allows **asynchronous tasks** (e.g., setTimeout, fetch) to run after synchronous code finishes.



It is a Small Representation on how ASYNCHRONOUS OPERATIONS work in JavaScript.