**SUMÁRIO**

[1. javascript from browser to server 1](#_Toc148723874)

[1.1. WHAT IS JAVASCRIPT 1](#_Toc148723875)

[1.2. BROWSERS AND JS ENGINE 1](#_Toc148723876)

[2. nodejs – a runtime environment 1](#_Toc148723877)

[3. FEATURES 2](#_Toc148723878)

[3.1. JUST IN TIME (JIT) 2](#_Toc148723879)

[3.2. SINGLE THREAD 2](#_Toc148723880)

[3.3. NON-BLOCKING I/O STRATEGY 2](#_Toc148723881)

# javascript from browser to server

## WHAT IS JAVASCRIPT

JavaScript is primarily a client-side language. JavaScript started at Netscape, a web browser developed in the 1990s. A webpage can contain embedded JavaScript, which executes when a user visits the page. The language was created to allow web developers to embed executable code on their webpages, so that they could make their webpages interactive, or perform simple tasks. Today, browser scripting remains the main use-case of JavaScript.

JavaScript is an interpreted language, not a compiled language. A program such as C++ or Java needs to be compiled before it is run. The source code is passed through a program called a compiler, which translates it into bytecode that the machine understands and can execute. In contrast, JavaScript has no compilation step. Instead, an interpreter in the browser reads over the JavaScript code, interprets each line, and runs it.

## BROWSERS AND JS ENGINE

**The JavaScript engine is simply a computer program that interprets JavaScript code.** The engine is responsible for executing the code. Every major browser has a JavaScript engine that executes JavaScript code. The most popular one is the Google Chrome [V8](https://en.wikipedia.org/wiki/JavaScript_engine) engine.

**When JavaScript executes within a web browser it is operating within the browser's runtime environment**. Think of the JavaScript runtime as the house with all the components needed to run JavaScript.  The browser runtime environment provides additional features to JS, such as access to the DOM which enables interaction with web page elements, handling events, and manipulating the page structure.

# nodejs – a runtime environment

**Node.js provides a server-side runtime environment for executing JavaScript outside the browser**. Because it executes JavaScript outside the browser, it does not have access to the web APIs. Instead, the Node.js runtime environment replaces it with something called C++ bindings and the thread pool.

Furthermore, the NodeJS uses the V8 engine, the core of Google Chrome. This allows it to be very powerful and optimized.

# FEATURES

## JUST IN TIME (JIT)

JavaScript used to be an interpreted language, but interpreted languages are slower compared to compiled languages.

In order to optimize the performance of web applications, JavaScript combines both compilation and interpretation. This is called Just-in-Time compilation. This method compiles the entire code into machine code all at once and executes it.

Just-in-Time compilation involves the same two processes as regular compilation, but here the machine code isn’t written into a binary file. The code is also executed right away after compilation.

## SINGLE THREAD

Node.js is known to be a single-threaded runtime environment, meaning that a program’s code is executed entirely line after line. In a multithreaded process, the processor can switch execution resources between threads, resulting in concurrent execution. Concurrency indicates that more than one thread is making progress, but the threads are not actually running simultaneously.

## NON-BLOCKING I/O STRATEGY

**Blocking I/O operations are synchronous**, meaning they block the execution of the program until the I/O operation is completed. This means that if a thread is performing a blocking I/O operation, it can't do anything else until that operation is finished.

**Non-blocking I/O operations are asynchronous**, which means they do not block the execution of the program. Instead of waiting for the I/O operation to complete, Node.js initiates the operation and then moves on to the next task without waiting. This allows Node.js to handle thousands of simultaneous connections to a single server without introducing the burden of row concurrency management. However, JavaScript execution in Node.js is single threaded. So concurrency only refers to the **event loop's ability to execute callback** functions after completing any other processing.