Introduction to JavaScript and Its Applications

Generated: 2025-10-21 12:39 PM

SUMMARY:

JavaScript is a high-level, single-threaded, garbage-collected language with a non-blocking event loop. Created in 1995 by Brendan Eich, it was originally named Moka but later renamed to sound like Java. Today, it's a fully-featured language used for building front-end web applications and server-side applications with Node.js.

KEY MOMENTS:

00:01:00 - JavaScript was created by Brendan Eich in 1995

00:02:30 - JavaScript is the only language other than web assembly natively supported in browsers

00:04:00 - JavaScript can be used for server-side applications with Node.js, mobile applications with React Native or Ionic, and desktop apps with Electron

VISUAL ANALYSIS:

Analyzed 6 frames

00:00 - The image is a digital art piece featuring an abstract design with geometric shapes, including a cube at the center that is lit up with neon yellow light. To the right of the central cube, there appears to be a stylized representation of a face or mask made from lines and shapes in white.

00:30 - The image displays a colorful abstract pattern that appears to be a digital or graphic illustration with various shapes and symbols, including what looks like a monitor displaying a character and a keyboard. It seems to have a retro-futuristic theme.

01:01 - The image shows a black and white circular logo with the words "EVENTLOOP" in white and "180 degrees of freedom" in smaller font at the top. The graphic includes concentric circles with a darker tone towards the center.

Text: "EVENT LOOP"

01:32 - The image shows a computer screen with a user interface featuring a text editor window. There is a button labeled "Button me," followed by a text field that says "Documentation". Below this text field, there are two buttons: "Click me," which appears to be inactive, and "Grab me," which seems to be selected as the focus of attention. The user interface has a theme with a dark background and light text.

Text: "JS app,js appJjs q) document.querySelector(â€˜buttonâ€™) ; | click me GRAB ME"

02:03 - The image displays a computer screen showing a code editor with highlighted lines of code. A yellow "error" message is prominently displayed at the top center, indicating that there

has been an error in the code execution. The error message is "Humanoid constructor not found." Below this message, the error is described as "Class 'humanoid' constructor not found," with a path to its location in the file system provided: "/usr/local/include/humanoid.h". A log file is shown to contain the specific details of the error, including the command "make", and an indication that the "walk" function was being executed before the error occurred.

Text: "JS app.js appjs\% Humanoid +) class Humanoid { constructor() { this.dna = 'S'; } | walk() { | console.log('going for a walk...') }"

02:34 - The image features a computer screen displaying an icon of a fire with a caption "Freship".

Text: "fireship.io"

TRANSCRIPT:

Javascript - a high-level single-threaded garbage collected, interpreted, or just in time-compiled, prototype-based multi-paradigm dynamic language with a non-blocking event loop, made famous for building websites. It was created in 1995 in just one week by Brendan I, with the goal of adding an easy-to-learn scripting language to the Netscape browser. It was originally named Moka, but the genius marketing people of the time wanted it to sound like that sexy new Java language. Today, it's a fully-featured language that continues to evolve through the ECMAScript standard. It's most well-known for building front-end web applications, because it's the only language other than web assembly that is natively supported in browsers. However, anything that can be built with Javascript will be built with Javascript. Like server-side applications with Node.js, mobile applications with React Native or Ionic, and desktop apps with Electron. It's an interpreted scripting language, but tools like the V8 engine and Chromium use a just-in-time compiler to convert it to machine code at runtime. It's also excellent at handling IO-intensive jobs, despite the fact that it's a single threaded language made possible by a non-blocking event loop that can queue up work in the background without blocking the main thread. To get started, create a file ending in .js. Your code will start executing from the global context, use the console to log a value with the built-in debugger. Now, think about where you want to run this file. Is it a front-end browser or a backend Node.js server? In the browser, you can reference it with a script tag, then the browser will execute it when you open that HTML file. On a website, Javascript is often used to grab an element from the DOM. Document query selector will grab the first button, then we can assign it to a variable with either var, let, or const. Bar is the OG way to do it, but is typically avoided. Let is for variables that can be reassigned, while const is for variables that cannot be reassigned. Now, to make the button interactive, we can add an event listener to it by defining a function for the on-click event, we can use the function keyword here, or the more concise arrow syntax. Now the event loop will execute this function whenever a new click occurs. Functions are first-class objects to support functional programming patterns, but Javascript also supports classes and inheritance for object-oriented patterns. Even though it's single-threaded, it can do work asynchronously with the promise API, which also supports the a-sync-wait syntax. JS code can also run on the server thanks to the Node.js runtime. Instead of buttons on a web page, it interacts with things like the file system API. Execute your code on the server by pulling up the terminal and entering the Node command. This has been Javascript in 100 seconds. Hit the like button and subscribe if you want to see more short videos like this, and become a FireShip Pro member if you want to learn how to do all kinds of crazy stuff with Javascript. Thanks for watching, and I will see you in the next one.

KEY TAKEAWAYS:

• JavaScript is a versatile language that can be used for various purposes

- Its non-blocking event loop makes it suitable for handling IO-intensive jobs
- It's also excellent at supporting functional programming patterns through first-class functions