PROSENSIA PVT LTD

M FAROOQ SAJID
INTRODUCTION TO JS

1. Introduction to JavaScript

JavaScript is a high-level, versatile programming language primarily used for adding interactivity, logic, and functionality to web pages. It is one of the **core technologies of the web**, alongside **HTML** and **CSS**.

Originally developed by **Brendan Eich** at Netscape in 1995, JavaScript has evolved from a simple scripting tool into a full-fledged language that now supports complex applications, servers, mobile development, and more.

Key Characteristics of JavaScript:

Interpreted (executed line-by-line by the browser)

Dynamic (no need to define types explicitly)

Event-driven (responds to user interactions)

Object-oriented (supports object-based design)

Prototype-based inheritance (unlike classical OOP)

---

2. Role of JavaScript in Web Development

Layer	Role of JS
HTML	Structure of the page
CSS	Styling and layout
JavaScript	Behavior and interactivity

JavaScript enables features such as:

Form validation

Dynamic content updates

Interactive maps

Sliders, animations, and popups

API integration and AJAX requests

---

3. How JavaScript Runs in the Browser

Step-by-Step Execution Process:

Page Load

When you visit a website, the browser downloads the HTML, CSS, and JS files.

HTML Parsing

The browser starts parsing the HTML and builds a **DOM (Document Object Model)** tree.

If a `<script>` tag is encountered:

It pauses HTML parsing

Loads and executes the JavaScript

Resumes HTML parsing

JavaScript Engine

The browser uses a **JavaScript engine** (e.g., Chrome's **V8**, Firefox's **SpiderMonkey**) to execute the code.

Code is converted into **machine-readable instructions** and executed line-by-line (interpretation).

Execution Context

JS runs inside the **Global Execution Context** (default environment).

Each function call creates a **new Execution Context**, forming the **Call Stack**.

Event Loop & Callback Queue

JS handles asynchronous operations using the **Event Loop**:

Functions like `setTimeout()` or `fetch()` are pushed to a **callback queue**

When the call stack is empty, the event loop pushes callbacks into it

---

Visualization: Browser JavaScript Components

pgsql

CopyEdit

```
Browser
├── HTML Parser → DOM Tree
├── CSS Parser → CSSOM Tree
├── JavaScript Engine
│   ├── Call Stack
│   ├── Heap (memory)
│   └── Event Loop + Callback Queue
└── Render Engine → Paint the screen
```

---

4. JavaScript Placement in HTML

JavaScript can be placed:

Inline: inside an element’s attribute

Internal: within `<script>` tags

External: linked via `<script src="script.js"></script>`

Best Practice: Place external scripts **before closing** `</body>` tag or use `defer`/`async` attributes in `<head>`.

html

CopyEdit

```
<script src="main.js" defer></script>
```

---

5. Modern JS Features (ES6+)

`let`, `const` (block scope)

Arrow functions `() => {}`

Template literals: `` `Hello ${name}` ``

Modules: `import`, `export`

Promises & `async/await`

Destructuring, Spread/Rest syntax

6. Conclusion

JavaScript is the language that powers the modern web. From basic button clicks to real-time updates, it enables dynamic, interactive user experiences. Understanding **how it runs in the browser**—via the DOM, execution context, and the event loop—provides the foundation needed to build fast, scalable, and reliable front-end applications.