JAVASCRIPT:

1. Story of JavaScript | JavaScript History

Brief History of JavaScript

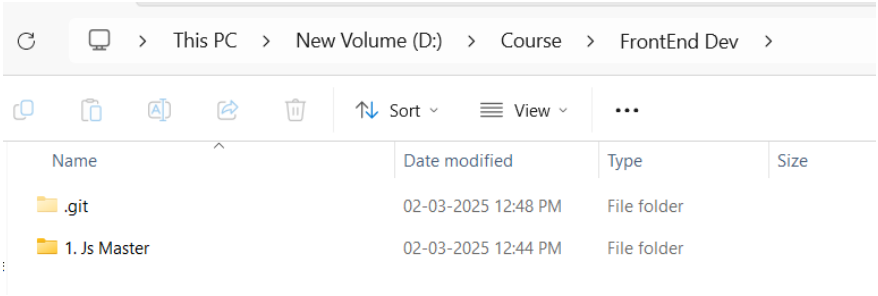
1. **1995 - Birth of JavaScript**
    - Created by **Brendan Eich (Brendon aaik)** at **Netscape** in just **10 days**.
    - First called **Mocha (moka)**, then **LiveScript**, and finally **JavaScript**.
2. **1996-1997 - Standardization**
    - Netscape gave JavaScript to **ECMA** (European Computer Manufacturers Association) for standardization.
    - **ECMAScript** (ES) was born in **1997**.
3. **2000s - Rise of AJAX & Web Apps**
    - JavaScript became powerful with **AJAX**, allowing web pages to update without reloading.
    - Used by big websites like **Google Maps & Gmail**.
4. **2009 - Node.js Revolution**
    - **Node.js** was created, allowing JavaScript to run on servers, not just browsers.
5. **2015 - ES6 & Modern JavaScript**
    - ES6 (ECMAScript 2015) introduced new features like **let/const, arrow functions, classes, promises**, etc.
    - JavaScript became more powerful & easier to use.
6. **Today**
    - JavaScript is used everywhere: **websites, mobile apps, servers, AI, games, and more**.
    - Popular frameworks: **React, Angular, Vue, Express, etc.**

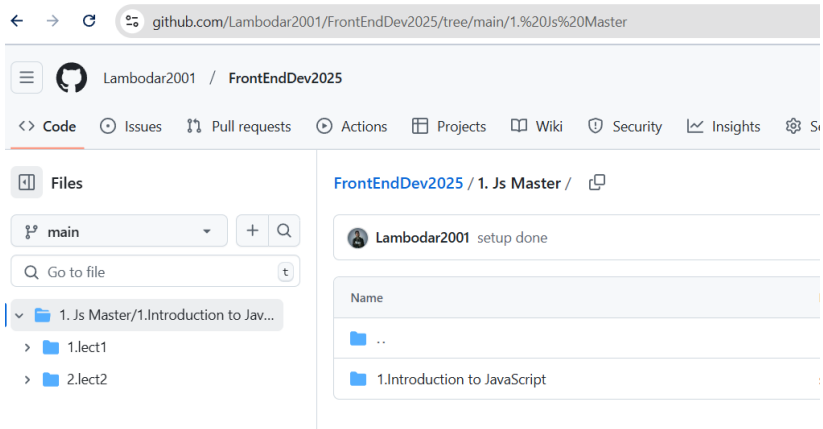
JavaScript is **primarily interpreted**, but modern JavaScript engines use **Just-In-Time (JIT) compilation**, making it a mix of both interpreted and compiled behaviour.

Explanation:

1. **Interpreted Nature:**
    - Originally, JavaScript was executed line-by-line by browsers using an interpreter.
    - This made it slower compared to compiled languages like C or Java.
2. **JIT Compilation (Modern Approach):**
    - Modern JavaScript engines (like V8 for Chrome, SpiderMonkey for Firefox) use JIT compilation.
    - Instead of purely interpreting JavaScript, the engine compiles frequently used code into **machine code** for better performance.

My notes, basics code file and github repo:





2. Ep.02: Introduction to JavaScript

1. Environment setup and run js file:

1. Run in chrome console

2. Run with node js engine which is extracted engine:

> use command to run js:
> node script.js

2. JavaScript Data Types (Short & Simple Notes)

1. Primitive Data Types (Basic & Immutable)

- **Number**

  `let num = 10;` (Integers, floats)

- **String**

  `let name = "Hello";` (Text, inside `""` or `''`)

Backtick Strings (Template Literals) in JavaScript

✅ **What is it?**
A modern way to write strings using backticks (`` ` ``) instead of quotes.

✅ **Why use it?**

- Allows **multi-line strings**
- Supports **variable interpolation** (${})
- Makes **string formatting easier**

```
console.log(greeting);

console.log(sum);
```

🚀 **Use backticks whenever you need dynamic or multi-line strings!**

✅ **Syntax & Examples:**

```
let name = "John";

let age = 25;


// 1. Multi-line string

let message = `Hello,

This is a multi-line string!`;


// 2. String interpolation

let greeting = `My name is ${name} and I am ${age} years old.`;


// 3. Expression inside

let sum = `5 + 5 = ${5 + 5}`;
```

- **Boolean**

  `let isTrue = true;` (Only `true` or `false`)

- **Undefined**

  `let x;` (Declared but not assigned)

- **Null**

  `let y = null;` (Empty, intentional absence of value)

- **BigInt**

  `let big = 123n;` (For very large numbers)

- **Symbol**

  `let sym = Symbol("unique");` (Unique identifier)

  7 primitive and one non primitive is object then it further divide into

2. Non-Primitive (Reference) Data Types

- **Object**

  `let obj = { key: "value" };` (Collection of properties)

- **Array**

  `let arr = [1, 2, 3];` (List of values)

- **Function**

  `function sayHello() {}` (Reusable block of code)

3. Type Conversion in JavaScript:

- **Implicit Conversion :** (Type Coercion (को'अश्न्/forcefully))
  (JavaScript automatically converts types)
- **Explicit Conversion :** (Manual Conversion)
  String Conversion:

✅ 1. Implicit Conversion (Type Coercion)

(JavaScript automatically converts types)

```
console.log("5" + 3);   // "53" (Number → String)
console.log("5" - 3);   // 2 (String → Number)
console.log(true + 1);  // 2 (Boolean → Number)
console.log("5" * "2"); // 10 (Both Strings → Numbers)
```

🚀 Golden Rule:

- **`+` keeps strings as strings.**
- **`-, *, /, %` try to convert to numbers.**
- **If conversion fails, result is `NaN`.**

📌 Simple Rules to Remember

1. **`+` with a string → Always results in a string (concatenation).**

```
console.log("5" + 3); // "53"  (Number converted to String)
console.log("Hello" + 5); // "Hello5"
```

2. **`-, *, /, %` try to convert strings to numbers.**

```
console.log("10" - "2"); // 8 (Both converted to numbers)
console.log("5" * "2"); // 10 (Both converted to numbers)
console.log("10" / 2); // 5 (String converted to number)
console.log("10" % 3); // 1 (String converted to number)
```

3. **If the string is no**

- Js useful commands

1. console.log('Hello, World!');
2. alert('Hello, World!');
3. typeof(var)