Chatper-1

Introduction

1. What is JavaScript?

JavaScript is a programming language. We use it to give instructions to the computer.

Definition: **JavaScript** is a high-level, interpreted **programming language** used primarily to create **interactive and dynamic content on websites**.

◻ Key Features of JavaScript:

- **Client-Side Language**: Runs directly in the browser (like Chrome, Firefox, etc.)
- **Lightweight & Flexible**: Easy to learn and very versatile
- **Interpreted**: Doesn’t need to be compiled—runs as is
- **Event-Driven**: Can respond to user actions (clicks, scrolls, etc.)
- **Object-Oriented**: Uses objects and functions for building reusable code
- **Multi-Paradigm**: Supports procedural, object-oriented, and functional styles

🔧 What Can JavaScript Do?

- Show/hide elements on click
- Validate form inputs
- Create image sliders, modals, and dropdowns
- Fetch data from APIs without reloading the page (AJAX)
- Build entire web apps (using frameworks like React, Vue, Angular)
- Power backend services (with **Node.js**)

```
<button onclick="sayHello()">Click Me</button>


<script>
 function sayHello() {
   alert("Hello, Sanjib!");
 }
</script>
```

☞ When you click the button, JavaScript shows a popup message.

Console and Console Panel:

console.log(); is used to print a message to the console.

Console.log("Sanjib Ghosh"); // ; is used to finish that coding line.

Output: Sanjib Ghosh

Html:

Html is used to create structure of the website.

🔗 Here's how it works:

1 ✅ HTML (HyperText Markup Language):

- It's the **foundation of all web pages**.
- It tells the browser **what to display**—like text, images, links, buttons, forms, etc.
- Think of it as the **skeleton** of a webpage.

2 □ The Browser's Role:

- The browser (like Chrome or Firefox) **reads your HTML**, interprets it, and **displays** the webpage on screen.
- It also handles other web development languages:
  - **CSS** for styling
  - **JavaScript** for interaction

3 🎯 In web development:

- You **write HTML** to structure content.
- You **run it in a browser** to **see the result**.
- The browser connects users to the website using your HTML as the base.

Basic Variable and Data types:

□ 1. Variables in JavaScript

◆ What is a Variable?

A **variable** is a container used to **store data** (like numbers, text, etc.).

◆ How to Declare Variables:

Since ES6 (modern JavaScript), we use:

- `let` – can be updated but not re-declared
- `const` – cannot be updated or re-declared
- `var` – old way, avoid using in modern code

```
let name = "Sanjib";
const age = 25;
var city = "Kolkata";
```

✓ JavaScript is a Dynamically Typed Language:

This means:

- **You don't need to declare the data type** of a variable explicitly (like `int`, `string`, etc.).
- JavaScript **automatically detects** the type **based on the assigned value** at runtime.

```
let x = 10;         // JavaScript understands this as a Number
let name = "Sanjib"; // JavaScript treats this as a String
let isStudent = true; // Boolean
```

You didn't tell JS the types — ***it figured them out automatically in Run-time i.e while running code in IDEs.***

That's why it's called loosely typed or dynamically typed — the variable type can change as the program runs.

Why it matters:

- Makes coding faster and more flexible.
- But you must be careful — **type-related bugs** can happen if you're not consistent.

□ 2. JavaScript Data Types

JavaScript has **two types** of data:

- **Primitive Types** (basic values)
- **Non-Primitive Types** (objects)

---

4 ◆ Primitive Data Types:

Data Type	Example	Description
String	`"Hello"` or `'Hi'`	Text inside quotes
Number	`25`, `3.14`, `-10`	Integer or float
Boolean	`true`, `false`	True or false
Null	`null`	Empty or no value
Undefined	`undefined`	Declared but no value assigned
Symbol	`Symbol('id')`	Unique identifier (advanced)
BigInt	`1234567890123456789 0n`	Large integers (advanced use)

```
let name = "Sanjib";          // String
let age = 25;                 // Number
let isStudent = true;         // Boolean
let emptyValue = null;          // Null   →type of null is object.
let notAssigned;              // Undefined By default all variable are Undefined until
assign any value within that.
```

---

5 ◆ Non-Primitive Data Types:

Type	Example
Object	`{ name: "Sanjib", age: 25 }`
Array	`[1, 2, 3]` or `["a", "b"]`
Function	`function greet() {}`

6 ✓ **Note**:

To check the data type of a variable, **use typeof:**

```
let value = 10;
console.log(typeof value);                    // "number"
```

- **Assignment operator :** In **Js** **=** is an assignment operator.

- **Variable naming rule :**

1. Case sensitive
2. Only letters, digits, _, $ is allowed.
3. Only a letter , or _ ,or $ should be the first character
4. Reserved word cannot be variable name. **Example- console.**

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Lexical_grammar#reserved_words

- Best_Convention: firstName, lastName,ageYear etc.

  1.fullName----Camel Case(✓)
  2.full_name----Snake Case

3.full-name------Kabab Case
4.FullName-------Pascale Case

3. Variable declaration Rule:

- We can declare variable like
  Age=32;
  isFollow=true;
  but it is not that proper way in JS.

var: Variable can be re-declared and updated. A global scope variable. (used before 2015)

let: Variable can't be re-declared but can be updated. A block scope variable.

const: Variable can't be re-declared or can't be updated. A block scope Variable.

At present **var** is not used – after ECMA script 6. **(also known as MODERN JS)**
New standard of Js.

```
// we can update instead of re-declaration
of variable.
let fullName="Sanjib Ghosh";
fullName="Sanjib Ghosal";
fullName="Sanjib Karuna";
console.log(fullName);
```

➢ `const` in JavaScript

When you declare a variable with `const`, it means:

- ❌ You **cannot reassign the variable** to a completely new value.
- ✅ But if the value is an **object or array**, you **can change the contents** (i.e. properties or elements) inside it.

```
const student1 = {
  fullName: "Vishal Biswas",
  age: 25,
  marks: 95,
  isPass: true
};

// These are all valid:
student1["fullName"] = "Debdip Das";
student1["age"] = student1["age"] - 3;
student1["marks"] = student1["marks"] - 10;
```

➢✅ This works because:

- You're **not changing the reference** to the object.
- You're **modifying the internal properties**, which is allowed.

➢ ❌ **What is NOT allowed:**

```
student1 = {
  fullName: "New Student"
};
```

🚫 This will throw an error:

```
TypeError: Assignment to constant variable.
```

Because you're trying to reassign the whole object, which `const` does not allow.

➢ ◻ Summary:

Action	Allowed with `const`?
Reassigning a new object	❌ No
Modifying properties of object	✓ Yes
Reassigning a new array	❌ No
Modifying array elements	✓ Yes