# Module : JavaScript

### JavaScript Introduction

**Question 1: What is JavaScript? Explain the role of JavaScript in web development.**

**Answer:**

* **JavaScript** is a high-level, interpreted, and lightweight programming language mainly used to make web pages **interactive and dynamic**.
* It is also known as the **scripting language of the web**, supported by all modern browsers.

**Role in Web Development:**

1. **Structure + Style + Behaviour**
   * **HTML** → provides the structure of a webpage.
   * **CSS** → provides styling and layout.
   * **JavaScript** → adds interactivity and functionality.
2. **Key Roles of JavaScript:**
   * Validating user input in forms (e.g., checking email format).
   * Creating dynamic content updates without reloading (e.g., live chat, search suggestions).
   * Manipulating HTML & CSS (e.g., showing/hiding elements, animations).
   * Working with APIs for data fetching (AJAX, Fetch API).
   * Building full-fledged web applications with frameworks like React, Angular, Vue.

**Question 2: How is JavaScript different from other programming languages like Python or Java?**

**Answer:**

| **Feature** | **JavaScript** | **Python** | **Java** |
| --- | --- | --- | --- |
| **Type** | Scripting language, mainly for web | General-purpose, data science, AI, scripting | Object-oriented, used for enterprise apps |
| **Execution** | Runs in browsers (client-side) & Node.js (server-side) | Runs on Python interpreter | Runs on JVM (Java Virtual Machine) |
| **Syntax** | C-like, lightweight | Very simple, readable (English-like) | Verbose, strict |
| **Use Case** | Web interactivity, front-end, full-stack | Data analysis, AI, automation | Enterprise apps, Android development |
| **Typing** | Dynamically typed | Dynamically typed | Statically typed (need to declare data types) |
| **Speed** | Faster in browsers (JIT compilation) | Slower compared to JS | Faster than JS & Python (compiled bytecode) |

* **JavaScript** = best for **web development**.
* **Python** = best for **AI, ML, data science**.
* **Java** = best for **enterprise applications & Android apps**.

**Question 3: Discuss the use of <script> tag in HTML. How can you link an external JavaScript file to an HTML document?**

Use of <script> tag in HTML:

* The <script> tag is used to **add JavaScript** to an HTML page.
* It can contain JavaScript code directly inside the HTML, or it can link to an external JavaScript file.
* Common uses: form validation, animations, DOM manipulation, and adding interactivity to webpages.

**Linking an External JavaScript File:**

* Save your JavaScript code in a separate file (e.g., script.js).
* Use the <script> tag with the src attribute to link it:

### Variables and Data Types

**Question 1: What are variables in JavaScript? How do you declare a variable using var, let, and const?**

**Answer:**

* A **variable** in JavaScript is a container used to store data values.
* Variables allow us to reuse and manipulate data in a program.

**Ways to declare variables:**

1. **var** → Old way, function-scoped, allows redeclaration & re-assignment.

var name = "abc";

1. **let** → Modern way, block-scoped, allows re-assignment but **not redeclaration**.

let age = 20;

age = 21; // ✅ allowed

1. **const** → Block-scoped, value cannot be changed (constant).

const pi = 3.14;

pi = 3.1416; ❌ Error

**Question 2: Explain the different data types in JavaScript. Provide examples for each.**

**Answer:**  
JavaScript has **two categories of data types**:

**1. Primitive Data Types (single value, immutable)**

* **String** → "Hello"
* let str = "Hello World";
* **Number** → 10, 3.14
* let Num = 25;
* **Boolean** → true / false
* let is\_active = true;
* **Undefined** → declared but not assigned a value
* let x;
* console.log(x); // undefined
* **Null** → intentional absence of value
* let y = null;

**2. Non-Primitive (Reference) Data Types**

* **Object** → collection of key-value pairs
* let person = { name: "Ayan", age: 20 };
* **Array** → ordered collection
* let fruits = ["apple", "banana", "mango"];
* **Function** → block of reusable code

function greet() { return "Hello"; }

**Question 3: Difference between Undefined and Null**

* **Undefined**: Variable declared but no value assigned (default).
* **Null**: Intentionally assigned empty value by the programmer

### JavaScript Operators

**Question 1: What are the different types of operators in JavaScript? Explain with examples.**

In JavaScript, operators are symbols used to perform operations on values and variables.

**1. Arithmetic Operators → used for mathematical calculations.**

let a = 10, b = 5;

console.log(a + b); // 15 (Addition)

console.log(a - b); // 5 (Subtraction)

console.log(a \* b); // 50 (Multiplication)

console.log(a / b); // 2 (Division)

console.log(a % b); // 0 (Modulus)

console.log(a \*\* 2); // 100 (Exponentiation)

**2. Assignment Operators → used to assign values.**

let x = 10;

x += 5; // x = x + 5 → 15

x -= 3; // x = x - 3 → 12

x \*= 2; // x = x \* 2 → 24

x /= 4; // x = x / 4 → 6

**3. Comparison Operators → used to compare values (returns true/false).**

let a = 10, b = 20;

console.log(a == b); // false (equal to, checks value only)

console.log(a === b); // false (strict equal, checks value + type)

console.log(a != b); // true (not equal)

console.log(a > b); // false

console.log(a < b); // true

console.log(a >= 10); // true

console.log(b <= 20); // true

**4. Logical Operators → used with Boolean values.**

let p = true, q = false;

console.log(p && q); // false (AND → both true required)

console.log(p || q); // true (OR → at least one true)

console.log(!p); // false (NOT → reverses the value)

**Question 2: What is the difference between == and === in JavaScript?**

* == → checks only value (loose equality).
* === → checks value **and** type (strict equality, preferred in modern JS).

### Control Flow (If-Else, Switch)

**Question 1: What is control flow in JavaScript? Explain how if-else statements work with an example.**

**Answer:**

* **Control flow** is the order in which statements in a program are executed.
* By default, JavaScript executes code from **top to bottom**.
* Using conditional statements like **if-else**, we can control which block of code runs based on conditions.

**How if-else works:**

* if → runs code if condition is true.
* else if → checks another condition if the first is false.
* else → runs if all conditions are false.

**Example:**

let age = 18;

if (age < 13) {

console.log("You are a child");

} else if (age >= 13 && age < 18) {

console.log("You are a teenager");

} else {

console.log("You are an adult");

}

// Output: You are an adult

**Question 2: Describe how switch statements work in JavaScript. When should you use a switch statement instead of if-else?**

**Answer:**

* A **switch statement** is used to perform different actions based on multiple possible values of a variable.
* It’s often used as a cleaner alternative to writing many if-else if statements.

**Syntax:**

let day = 3;

switch (day) {

case 1:

console.log("Monday");

break;

case 2:

console.log("Tuesday");

break;

case 3:

console.log("Wednesday");

break;

default:

console.log("Invalid day");

}

// Output: Wednesday

**When to use switch over if-else?**

* Use **if-else** → when checking **ranges or complex conditions**.
* Use **switch** → when checking **one variable against many fixed values** (e.g., days, menu options, status codes).