JavaScript Fundamentals

JavaScript is a high-level, interpreted, dynamically typed programming language widely used for web development. As one of the core technologies of the web, alongside HTML and CSS, JavaScript enables developers to create interactive, responsive, and dynamic applications.

---

1. Variables and Data Types

A variable serves as a storage unit for data, and JavaScript supports both primitive and non-primitive data types.

1.1 Primitive Data Types (Immutable)

Primitive data types represent single, immutable values.

- **String – Represents textual data enclosed in single, double, or template literals.**
- **Number – Represents both integer and floating-point values.**
- **Boolean – Represents logical values: true or false.**
- **Undefined – Denotes a variable that has been declared but not assigned a value.**
- **Null – Represents an explicitly assigned empty or unknown value.**
- **Symbol (ES6) – Used to create unique and immutable identifiers.**
- **BigInt (ES11) – Enables representation of arbitrarily large numbers beyond the Number type's limitations.**

1.2 Non-Primitive Data Types (Reference)

Non-primitive data types are mutable and store complex data structures.

- **Object – A collection of key-value pairs used to structure data.**
- **Array – An ordered list of values.**
- **Function – A reusable block of code, treated as a first-class object in JavaScript.**

---

2. Operators and Expressions

Operators are symbols that perform computations or logical evaluations on values and expressions.

2.1 Types of Operators

- **Arithmetic Operators – Perform mathematical calculations such as addition, subtraction, multiplication, and division.**
- **Comparison Operators – Compare values and return a Boolean result.**
- **Logical Operators – Used to combine or manipulate Boolean values.**
- **Assignment Operators – Assign values to variables (=, +=, -=, etc.).**
- **Ternary Operator – A shorthand conditional operator for concise if-else logic.**

---

3. Control Flow Statements

Control flow structures determine how code is executed based on specified conditions.

- **if-else Statement – Executes different blocks of code depending on a condition's Boolean evaluation.**
- **switch Statement – Evaluates an expression and executes a matching case.**

---

4. Variable Declarations: var, let, const

JavaScript provides three distinct keywords for variable declaration, each with different scope and behavior.

Declaration	Scope	Hoisting	Reassignment	Redeclaration
var	Function	Hoisted with undefined	Allowed	Allowed
let	Block	Hoisted without initialization	Allowed	Not Allowed
const	Block	Hoisted without initialization	Not Allowed	Not Allowed

Scope defines a variable's accessibility within different parts of a program.

---

5. Hoisting in JavaScript

Hoisting is JavaScript's behavior of moving declarations to the top of their respective scope before code execution.

- **Variables declared with var are hoisted but initialized as undefined.**
- **Variables declared with let and const are hoisted but remain in the Temporal Dead Zone (TDZ) until explicitly assigned a value.**
- **Function declarations are hoisted alongside their definitions, making them accessible before their declaration in the code.**

---

6. JavaScript Execution Context

JavaScript code executes in two distinct phases:

1. **Creation Phase**
    - **The Global Execution Context (GEC) is initialized.**
    - **Variables declared with var are hoisted and assigned undefined.**
    - **Function declarations are hoisted with their definitions.**
2. **Execution Phase**
    - **The script executes line by line.**
    - **Variables are assigned their respective values.**
    - **Functions are executed when invoked.**

---

7. Summary

- **JavaScript is a dynamically typed, interpreted programming language primarily used for web development.**
- **Variables store data and can be declared using var, let, or const.**
- **Data types include primitive types (string, number, boolean, etc.) and reference types (objects, arrays).**
- **Operators perform computations and logical evaluations.**
- **Conditional statements (if-else, switch) control program execution.**
- **Scope defines variable accessibility (global, function, block).**
- **Hoisting moves variable and function declarations to the top of their scope.**
- **JavaScript execution occurs in a two-phase process: creation and execution.**

This foundational knowledge serves as the basis for understanding more advanced JavaScript concepts, including closures, asynchronous programming, and event-driven architecture.