1. Arithmetic Operators

Arithmetic operators perform mathematical operations on numbers.

Operator	Description	Example	Result
`+`	Addition	`10 + 5`	`15`
`-`	Subtraction	`10 - 5`	`5`
`*`	Multiplication	`10 * 5`	`50`
`/`	Division	`10 / 5`	`2`
`%`	Modulus (remainder)	`10 % 3`	`1`
`**`	Exponentiation	`2 ** 3`	`8`
`++`	Increment	`let a = 5;` `a++;`	`a` becomes `6`
`--`	Decrement	`let b = 5;` `b--;`	`b` becomes `4`

2. Assignment Operators

Assignment operators assign values to JavaScript variables.

Operator	Example	Same As

`=`	`x = 5`	`x = 5`
`+=`	`x += 5`	`x = x + 5`
`-=`	`x -= 5`	`x = x - 5`
`*=`	`x *= 5`	`x = x * 5`
`/=`	`x /= 5`	`x = x / 5`
`%=`	`x %= 5`	`x = x % 5`
`**=`	`x **= 5`	`x = x ** 5`

3. Comparison Operators

Comparison operators compare two values and return a boolean (`true` or `false`) result.

Operator	Description	Example
`==`	Equal to (value only)	`5 == "5"` evaluates to `true`

`===`	Strict equal to (value and type)	`5 === "5"` evaluates to `false`
`!=`	Not equal to (value only)	`5 != "10"` evaluates to `true`
`!==`	Strict not equal to (value and type)	`5 !== "5"` evaluates to `true`
`>`	Greater than	`10 > 5` evaluates to `true`
`<`	Less than	`10 < 5` evaluates to `false`
`>=`	Greater than or equal to	`10 >= 10` evaluates to `true`
`<=`	Less than or equal to	`10 <= 5` evaluates to `false`

4. String Operators

JavaScript has a special operator for string concatenation.

Operator	Description	Example
`+`	Concatenation	`"Hello" + "World"`
`+=`	Concatenation assignment	`text += "World"`

5. Logical Operators

Logical operators are used to combine conditional statements.

Operator	Description	Example
`&&`	Logical AND	`(x > 5 && y < 10)`
`		`
`!`	Logical NOT	`!(x > 5)`

6. Bitwise Operators

Bitwise operators perform operations on the binary representation of numbers. These are less commonly used in general web development but are important for low-level operations.

Operator	Description	Example	Result (Decimal)
`&`	AND	`5 & 1`	`1` (0101 & 0001 = 0001)
`	`	OR	`5
`^`	XOR	`5 ^ 1`	`4` (0101 ^ 0001 = 0100)
`~`	NOT	`~5`	`-6` (inverts all bits)
`<<`	Left Shift	`5 << 1`	`10` (0101 << 1 = 1010)

`>>`	Right Shift	`5 >> 1`	`2` (0101 >> 1 = 0010)
`>>>`	Zero-fill Right Shift	`5 >>> 1`	`2` (same as `>>` for positive numbers)

7. Ternary Operator (Conditional Operator)

The ternary operator is a shorthand for an `if-else` statement. It takes three operands: a condition, an expression to execute if the condition is true, and an expression to execute if the condition is false.

Syntax	Description
`condition ? expressionIfTrue : expressionIfFalse`	Returns `expressionIfTrue` if `condition` is `true`, otherwise returns `expressionIfFalse`.

8. Type Operators

Type operators are used to determine the type of a variable or an operand.

Operator	Description	Example
`typeof`	Returns the type of a variable.	`typeof "hello"` returns `"string"`
`instanceof`	Returns `true` if an object is an instance of a specified object type.	`[1, 2, 3] instanceof Array` returns `true`