JavaScript - Beginner's Complete Guide

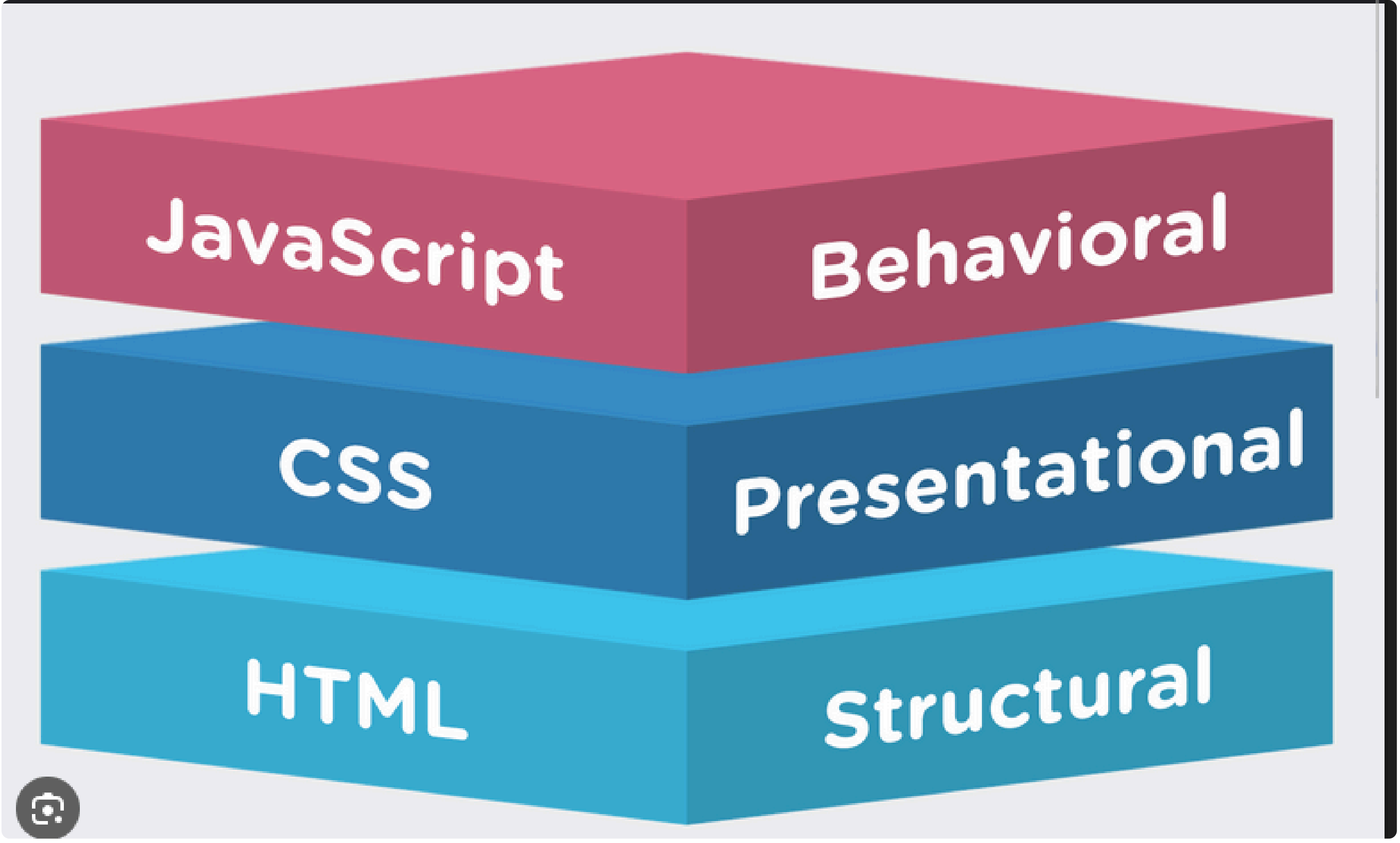
Table of Contents

What is JavaScript?

JavaScript is a scripting language that makes web pages interactive. It's the only programming language that runs natively in web browsers and is essential for modern web development.

Key Facts:

- **Client-side**: Runs in the user's browser
- **Dynamic**: Can change content without page reloads
- **Event-driven**: Responds to user interactions
- **Versatile**: Also runs on servers (Node.js)



Getting Started

Where to Write JavaScript

1. Inline JavaScript

```
<button onclick="alert('Hello!')">Click me</button>
```

2. Internal JavaScript

```
<script>
  console.log("Hello, World!");
</script>
```

3. External JavaScript

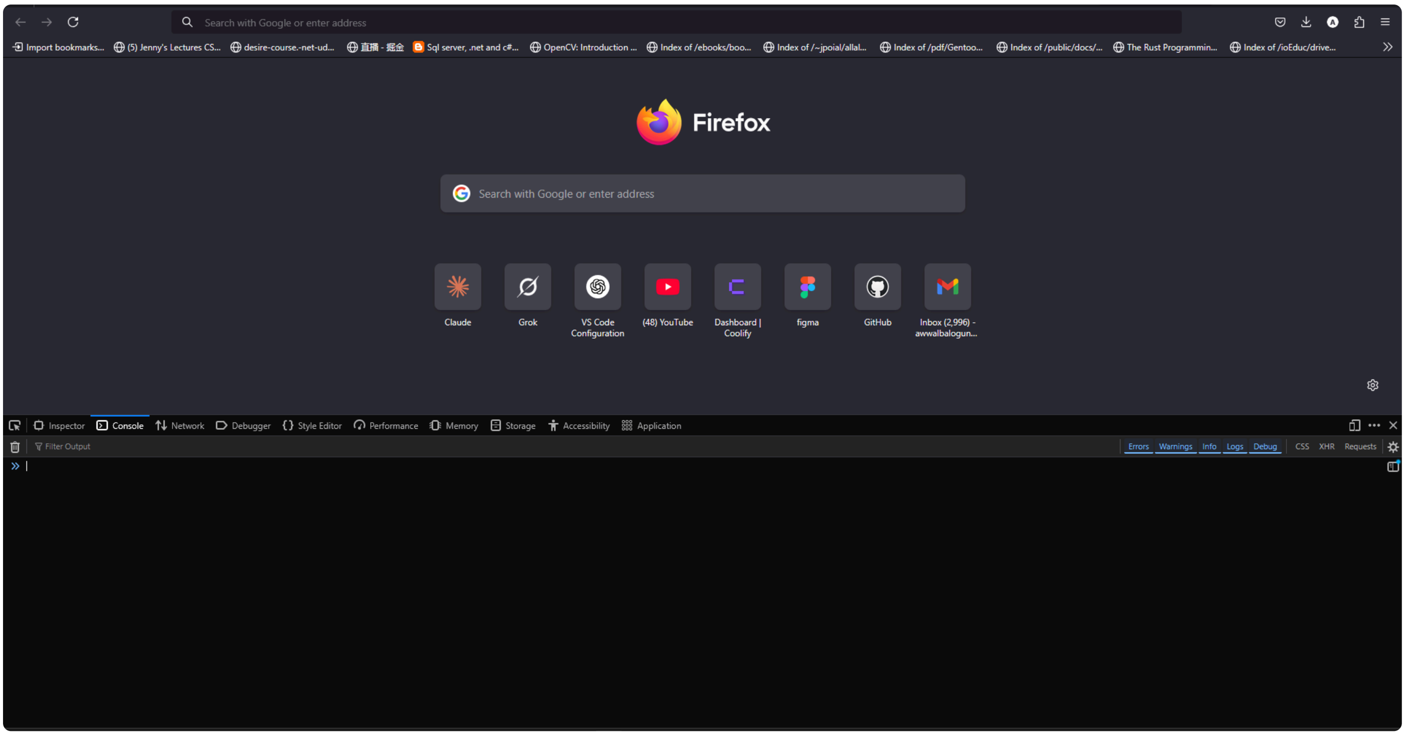
```
<script src="script.js"></script>
```

Your First JavaScript Program

```
// This is a comment
console.log("Hello, World!");
alert("Welcome to JavaScript!");
```

Browser Developer Tools

- **Chrome/Edge**: F12 or Ctrl+Shift+I
- **Firefox**: F12 or Ctrl+Shift+K
- **Safari**: Cmd+Option+I



Variables and Data Types

Declaring Variables

Modern Way (ES6+)

```
let name = "John";         // Can be changed
const age = 25;            // Cannot be changed
var city = "New York";     // Old way (avoid when possible)
```

Variable Naming Rules

```
// Valid names
let firstName = "John";
let _private = "secret";
let $element = document.getElementById("myDiv");
let userAge2 = 25;

// Invalid names
// let 2users = 10;        // Can't start with number
// let first-name = "";    // Can't use hyphens
// let class = "";         // Can't use reserved words
```

Data Types

Primitive Types

1. String

```
let message = "Hello World";
let name = 'John Doe';
let template = `Hello ${name}`;  // Template literal we use bacticks for ths `` it is similar to your string formatting in c#
```

2. Number

```
let age = 25;
let price = 99.99;
let negative = -10;
```

3. Boolean

```
let isActive = true;
let isComplete = false;
```

4. Undefined

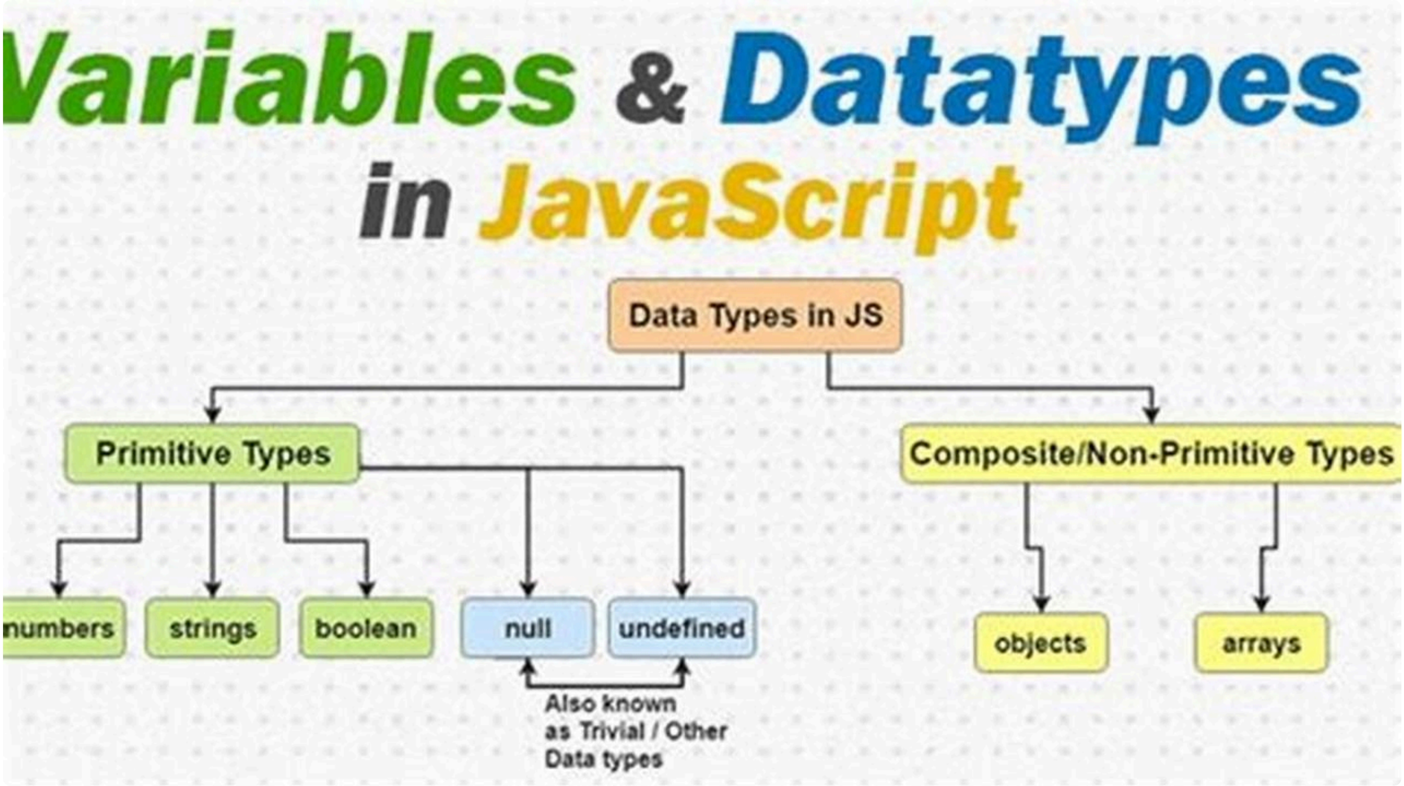
```
let notDefined;
console.log(notDefined); // undefined
```

5. Null

```
let empty = null;
```

Checking Data Types

```
// typeof is a keword in js used to check datatype it is similar to your .GetType() in c#
console.log(typeof "Hello");    // "string"
console.log(typeof 42);         // "number"
console.log(typeof true);       // "boolean"
console.log(typeof undefined);  // "undefined"
console.log(typeof null);       // "object" (this is a known quirk!)
```



Operators

Arithmetic Operators

```
let a = 10;
let b = 3;

console.log(a + b);  // 13 (Addition)
console.log(a - b);  // 7  (Subtraction)
console.log(a * b);  // 30 (Multiplication)
console.log(a / b);  // 3.333... (Division)
console.log(a % b);  // 1  (Remainder/Modulus)
console.log(a ** b); // 1000 (Exponentiation)
```

Comparison Operators

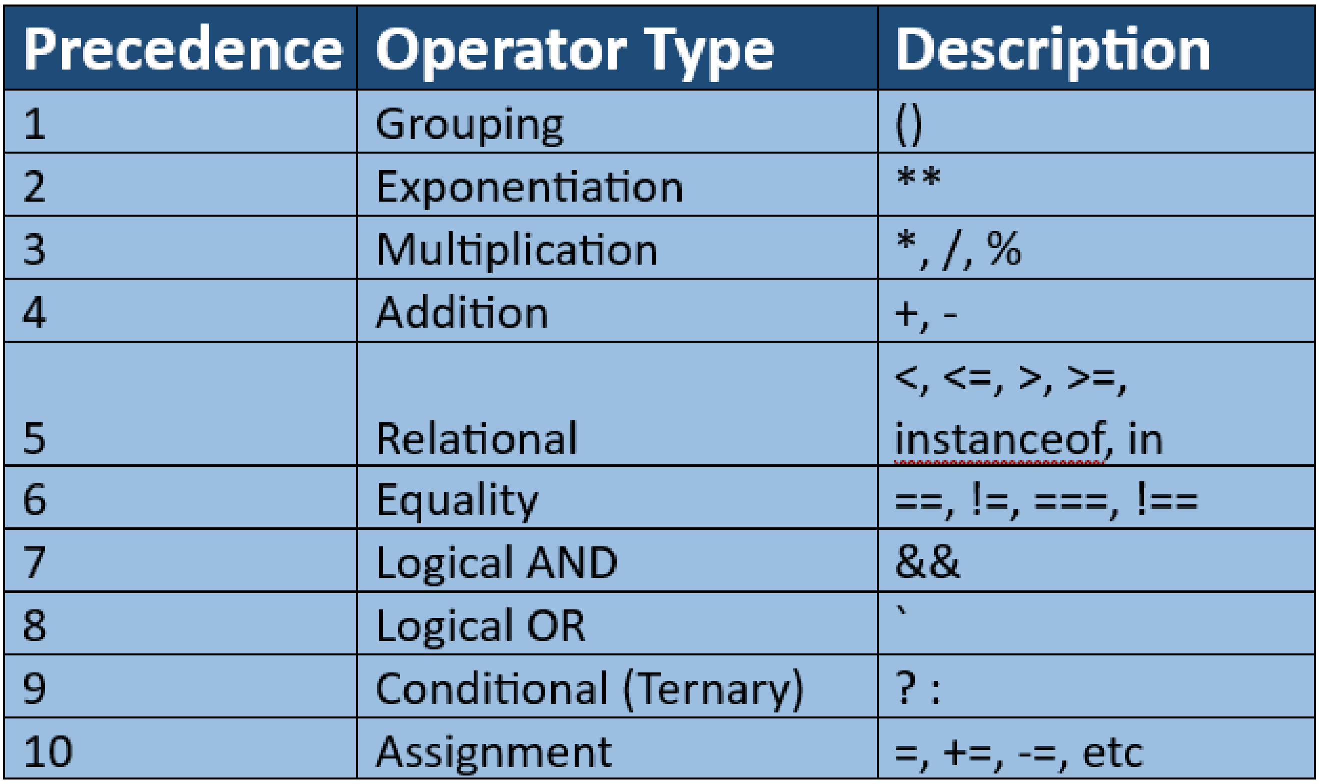
```
let x = 5;
let y = "5";

console.log(x == y);   // true  (Equal - converts types)
console.log(x === y);  // false (Strict equal - no conversion)
console.log(x != y);   // false (Not equal)
console.log(x !== y);  // true  (Strict not equal)
console.log(x > 3);    // true  (Greater than)
console.log(x >= 5);   // true  (Greater than or equal)
console.log(x < 10);   // true  (Less than)
console.log(x <= 5);   // true  (Less than or equal)
```

Logical Operators

```
let isAdult = true;
let hasLicense = false;

console.log(isAdult && hasLicense); // false (AND)
console.log(isAdult || hasLicense); // true  (OR)
console.log(!isAdult);              // false (NOT)
```

Assignment Operators

```
let score = 10;

score += 5;    // score = score + 5  (15)
score -= 3;    // score = score - 3  (12)
score *= 2;    // score = score * 2  (24)
score /= 4;    // score = score / 4  (6)
score++;       // score = score + 1  (7)
score--;       // score = score - 1  (6)
```



Precedence	Operator Type	Description
1	Grouping	()
2	Exponentiation	**
3	Multiplication	*, /, %
4	Addition	+, -
5	Relational	<, <=, >, >=, instanceof, in
6	Equality	==, !=, ===, !==
7	Logical AND	&&
8	Logical OR	`
9	Conditional (Ternary)	? :
10	Assignment	=, +=, -=, etc

Functions

Function Declaration

```
function greet(name) {
    return "Hello, " + name + "!";
}
```

```
let message = greet("Awwal");
console.log(message); // "Hello, Awwal!"
```

Function Expression

```
const greet = function(name) {
    return "Hello, " + name + "!";
};
```

Arrow Functions (ES6+)

```
// Single parameter, single expression
const square = x => x * x;

// Multiple parameters
const add = (a, b) => a + b;

// Multiple lines
const greetFormal = name => {
    const greeting = "Good day, " + name;
    return greeting + "!";
};
```

Parameters and Arguments

```
// Default parameters
function greet(name = "Guest") {
    return "Hello, " + name + "!";
}

console.log(greet());        // "Hello, Guest!"
console.log(greet("Awwal"));  // "Hello, Awwal!"

// Rest parameters
function sum(...numbers) {
    return numbers.reduce((total, num) => total + num, 0);
}

console.log(sum(1, 2, 3, 4)); // 10
```

Control Structures

Conditional Statements

if/else

```
let age = 18;

if (age >= 18) {
    console.log("You are an adult");
} else if (age >= 13) {
    console.log("You are a teenager");
} else {
    console.log("You are a child");
}
```

Ternary Operator

```
let status = age >= 18 ? "adult" : "minor";
console.log(status);
```

Switch Statement

```
let day = "Monday";

switch (day) {
    case "Monday":
        console.log("Start of work week");
        break;
    case "Friday":
        console.log("TGIF!");
        break;
    case "Saturday":
    case "Sunday":
        console.log("Weekend!");
        break;
    default:
        console.log("Regular day");
}
```

Loops

for Loop

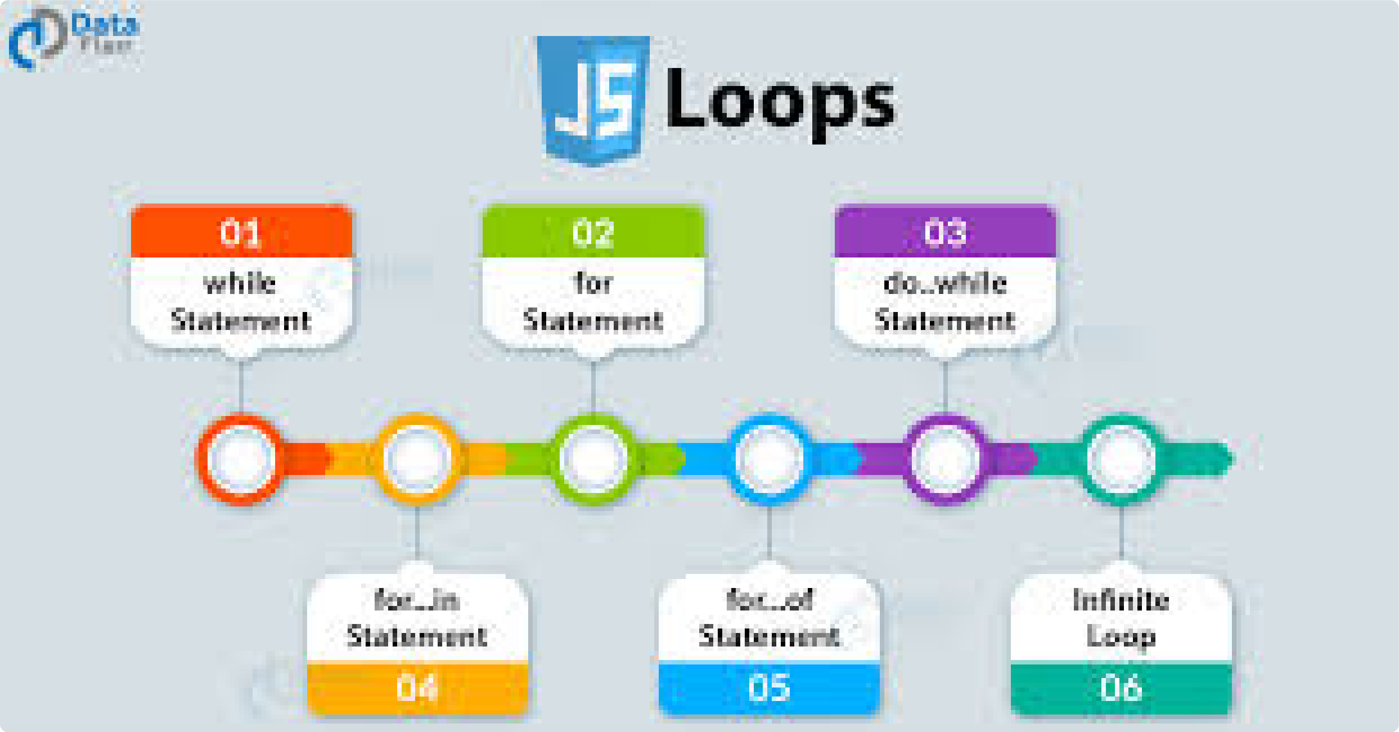
```
// Traditional for loop
for (let i = 0; i < 5; i++) {
    console.log("Count: " + i);
}

// for...of (for arrays)
let fruits = ["apple", "banana", "orange"];
for (let fruit of fruits) {
    console.log(fruit);
}

// for...in (for objects)
let person = {name: "Awwal", age: 30};
for (let key in person) {
    console.log(key + ": " + person[key]);
}
```

while Loop

```
let count = 0;
while (count < 3) {
    console.log("Count: " + count);
    count++;
}
```

do...while Loop

```
let input;
do {
    input = prompt("Enter 'quit' to exit:");
} while (input !== "quit");
```



Arrays(ArrayList)

note: Languages like python and js do not have the traditional arrays , they have what we call an arrayList

READ MORE HERE

`a dynamic array that allows you to store and manage collections of objects without explicitly specifying the type`

Creating Arrays

```
let fruits = ["apple", "banana", "orange"];
let numbers = [1, 2, 3, 4, 5];
let mixed = ["text", 42, true, null];
let empty = [];
```

Accessing Elements

```
let fruits = ["apple", "banana", "orange"];

console.log(fruits[0]);    // "apple"
console.log(fruits[1]);    // "banana"
console.log(fruits[-1]);   // undefined (no negative indexing)
console.log(fruits.length); // 3
```

Common Array Methods

Adding/Removing Elements

```
let fruits = ["apple", "banana"];

// Add to end
fruits.push("orange");        // ["apple", "banana", "orange"]

// Add to beginning
fruits.unshift("grape");      // ["grape", "apple", "banana", "orange"]

// Remove from end
let last = fruits.pop();      // "orange", array becomes ["grape", "apple", "banana"]

// Remove from beginning
let first = fruits.shift();   // "grape", array becomes ["apple", "banana"]
```

Array Iteration

```
let numbers = [1, 2, 3, 4, 5];

// forEach - execute function for each element
numbers.forEach(num => {
    console.log(num * 2);
});
```

```
// map - create new array with transformed elements
let doubled = numbers.map(num => num * 2);
console.log(doubled); // [2, 4, 6, 8, 10]

// filter - create new array with elements that pass test
let evenNumbers = numbers.filter(num => num % 2 === 0);
console.log(evenNumbers); // [2, 4]

// find - return first element that passes test
let found = numbers.find(num => num > 3);
console.log(found); // 4
```

Other Useful Methods

```
let fruits = ["apple", "banana", "orange"];

// Check if element exists
console.log(fruits.includes("banana")); // true

// Find index of element
console.log(fruits.indexOf("orange"));  // 2

// Join array into string
console.log(fruits.join(", "));         // "apple, banana, orange"

// Sort array
fruits.sort();
console.log(fruits); // ["apple", "banana", "orange"]
```

Objects

Creating Objects

```
// Object literal
let person = {
    name: "John Doe",
    age: 30,
    city: "New York",
    isEmployed: true
};

// Empty object
let empty = {};
```

Accessing Object Properties

```
let person = {
    name: "Awwal Muhammed",
    age: 30,
    "favorite color": "blue"
};

// Dot notation
console.log(person.name);     // "Awwal Muhammed"
console.log(person.age);      // 30

// Bracket notation
console.log(person["name"]); // "Awwal Muhammed"
console.log(person["favorite color"]); // "blue"

// Dynamic property access
let property = "age";
console.log(person[property]); // 30
```

Adding/Modifying Properties

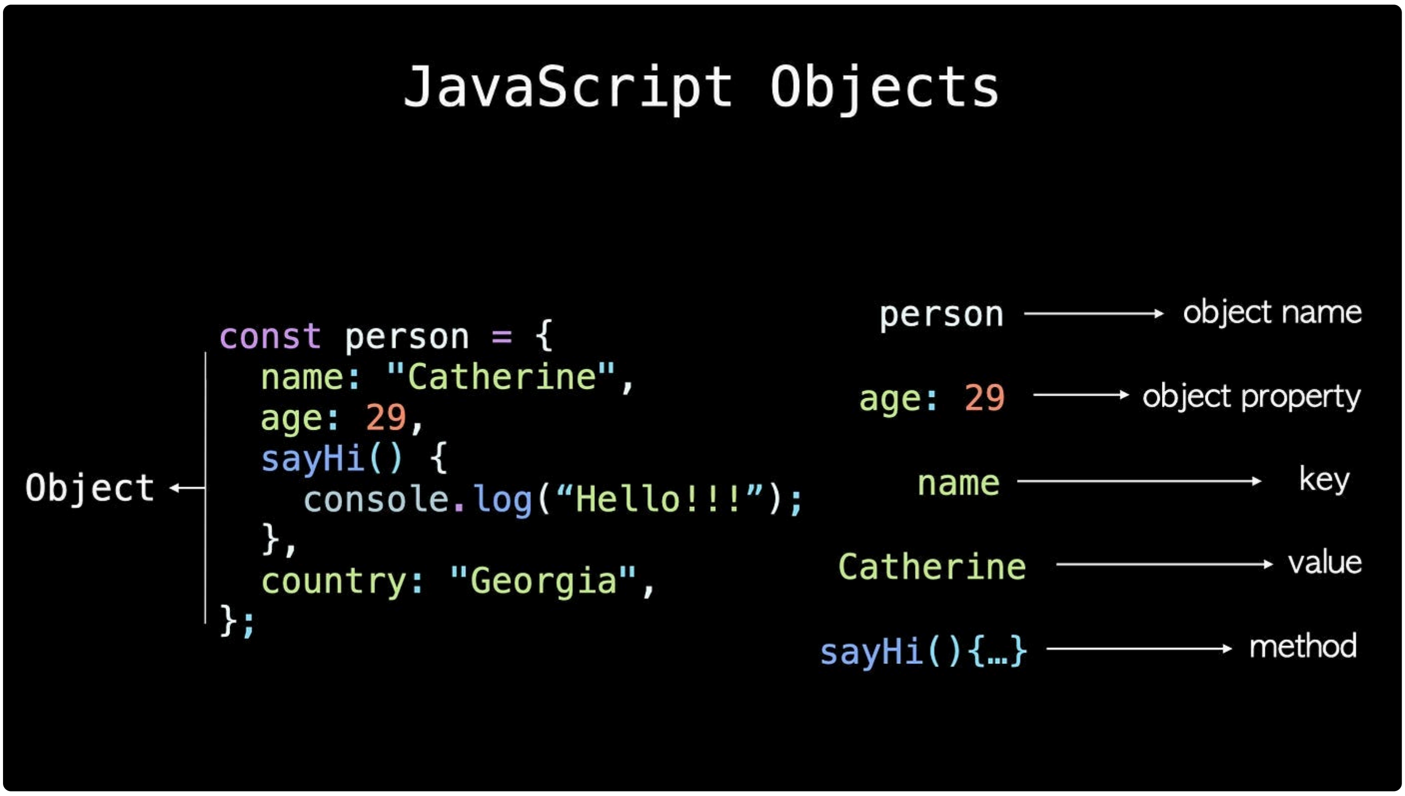
```
let person = {
    name: "Awwal"
};

// Add new property
person.age = 30;
person["city"] = "Boston";

// Modify existing property
person.name = "Balogun";

console.log(person); // {name: "Balogun", age: 30, city: "Boston"}
```

Object Methods

```
let calculator = {
    result: 0,

    add: function(num) {
        this.result += num;
        return this;
    },

    multiply: function(num) {
        this.result *= num;
        return this;
    },

    getValue: function() {
        return this.result;
    }
};

// Method chaining
let final = calculator.add(5).multiply(3).getValue();
console.log(final); // 15
```

Destructuring

```
let person = {
    name: "John",
    age: 30,
    city: "Boston"
};

// Extract properties into variables
let {name, age} = person;
console.log(name); // "John"
console.log(age);  // 30

// With different variable names
let {name: fullName, city: location} = person;
console.log(fullName); // "John"
console.log(location); // "Boston"
```



```
const person = {
  name: "Catherine",
  age: 29,
  sayHi() {
    console.log(“Hello!!!”);
  },
  country: "Georgia",
};
```

DOM Manipulation

Selecting Elements

```
// By ID
let element = document.getElementById("myId");

// By class name
let elements = document.getElementsByClassName("myClass");

// By tag name
let paragraphs = document.getElementsByTagName("p");

// Query selector (CSS-style)
let first = document.querySelector(".myClass");    // First match
let all = document.querySelectorAll(".myClass");  // All matches
```

Modifying Content

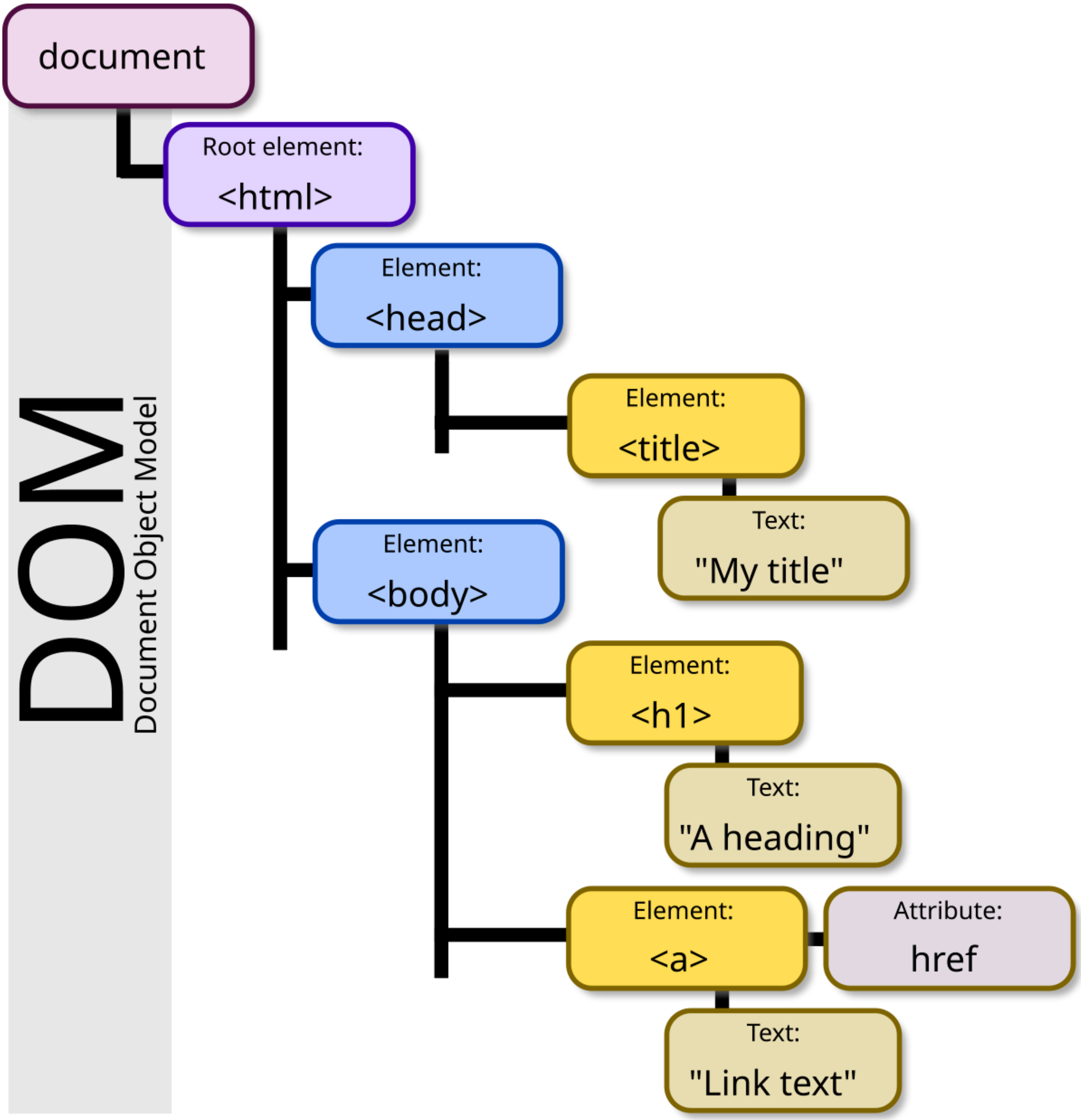
```
let element = document.getElementById("myDiv");

// Change text content
element.textContent = "New text content";

// Change HTML content
element.innerHTML = "<strong>Bold text</strong>";

// Change attributes
element.setAttribute("class", "newClass");
element.src = "newImage.jpg";  // For img elements
```

Styling Elements

```
let element = document.getElementById("myDiv");

// Individual styles
element.style.color = "red";
element.style.fontSize = "20px";
element.style.backgroundColor = "yellow";

// CSS classes
element.classList.add("newClass");
element.classList.remove("oldClass");
element.classList.toggle("active");
element.classList.contains("myClass"); // returns true/false
```

Creating and Adding Elements

```
// Create new element
let newDiv = document.createElement("div");
newDiv.textContent = "I'm a new div!";
newDiv.className = "dynamicDiv";

// Add to page
let container = document.getElementById("container");
container.appendChild(newDiv);

// Insert at specific position
container.insertBefore(newDiv, container.firstChild);
```



Events

Adding Event Listeners

```
// Method 1: addEventListener (recommended)
let button = document.getElementById("myButton");
```

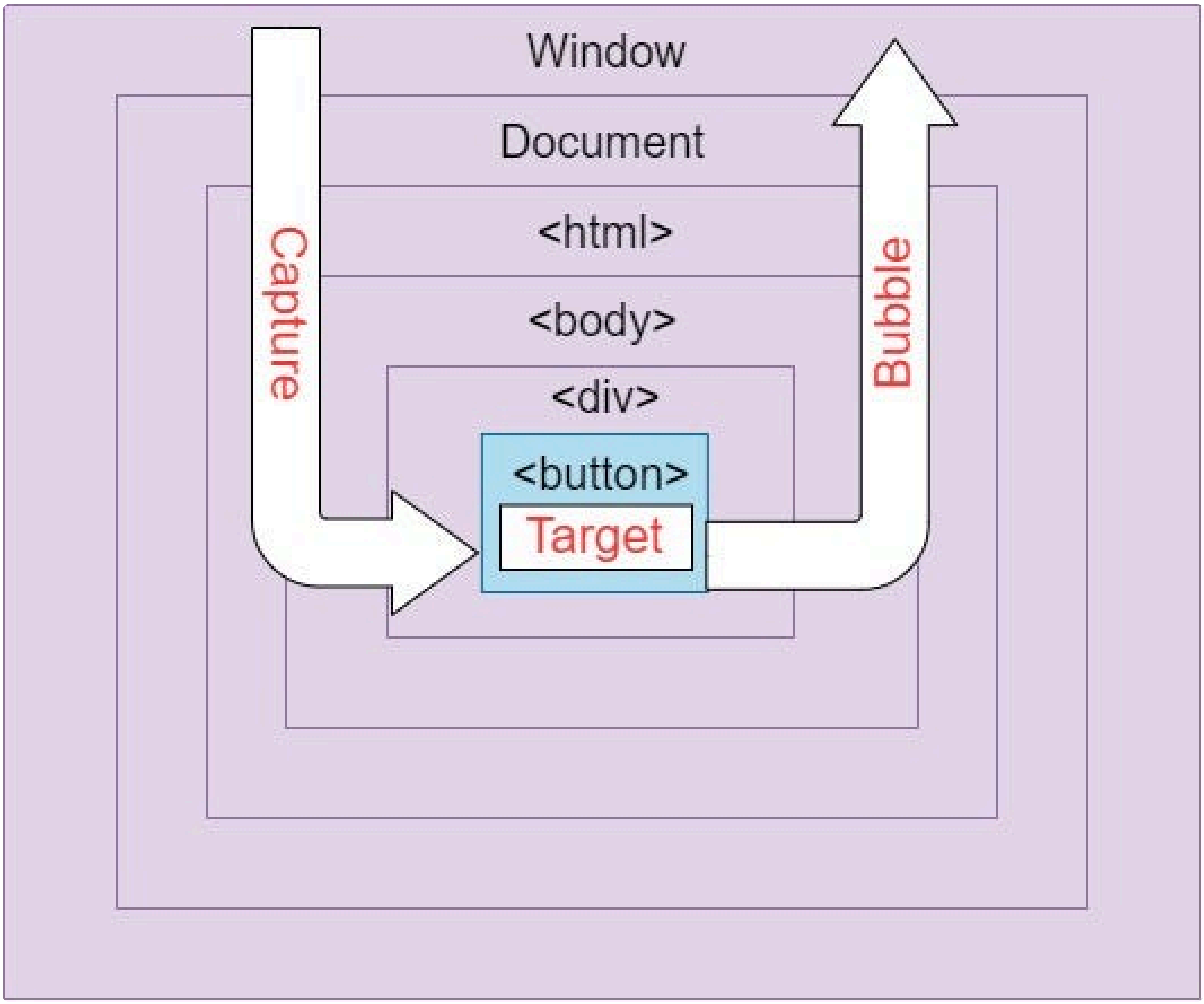
```
button.addEventListener("click", function() {
    alert("Button was clicked!");
});

// Method 2: Arrow function
button.addEventListener("click", () => {
    console.log("Button clicked!");
});

// Method 3: Named function
function handleClick() {
    console.log("Button clicked!");
}
button.addEventListener("click", handleClick);
```

Common Events

```
let input = document.getElementById("myInput");
let form = document.getElementById("myForm");

// Form events
input.addEventListener("focus", () => {
    console.log("Input focused");
});

input.addEventListener("blur", () => {
    console.log("Input lost focus");
});

input.addEventListener("input", (event) => {
    console.log("Input value:", event.target.value);
});

form.addEventListener("submit", (event) => {
    event.preventDefault(); // Prevent form submission
    console.log("Form submitted");
});

// Mouse events
button.addEventListener("mouseenter", () => {
    console.log("Mouse entered");
});

button.addEventListener("mouseleave", () => {
    console.log("Mouse left");
});

// Keyboard events
document.addEventListener("keydown", (event) => {
    console.log("Key pressed:", event.key);
});
```

Event Object

```
button.addEventListener("click", function(event) {
    console.log("Event type:", event.type);
    console.log("Target element:", event.target);
    console.log("Mouse position:", event.clientX, event.clientY);

    // Prevent default behavior
    event.preventDefault();

    // Stop event from bubbling up
    event.stopPropagation();
});
```



Asynchronous JavaScript

these links will asssit
[FireShip]()
[Youtube]()
[FreeCodeCamp]()

Common Patterns

Form Validation

```javascript
function validateForm() {
    let email = document.getElementById("email").value;
    let password = document.getElementById("password").value;
    let errors = [];

    // Email validation
    if (!email.includes("@")) {
        errors.push("Please enter a valid email");
    }

    // Password validation
    if (password.length < 8) {
        errors.push("Password must be at least 8 characters");
    }

    // Display errors
    let errorDiv = document.getElementById("errors");
    if (errors.length > 0) {
        errorDiv.innerHTML = errors.join("<br>");
        return false;
    } else {
        errorDiv.innerHTML = "";
        return true;
    }
}

document.getElementById("myForm").addEventListener("submit", function(event) {
    if (!validateForm()) {
        event.preventDefault();
    }
});
```

Further Reading

Javascript is not a language that can be mastered with simple eyes
here are some resources to help

1. Js in 100 seconds by fireship
2. Mosh Ahmedani
3. Harvard HTML,CSS,JS

`@TECHPYLON`