

Modul Tutorial JavaScript

Stevanus Denko Firdo Ananda – 72230633

Apa Itu JavaScript?

Definisi Umum

JavaScript (JS) adalah bahasa pemrograman tingkat tinggi, bersifat dinamis, dan berbasis prototipe yang digunakan untuk menciptakan interaktivitas dalam halaman web. JavaScript berjalan di sisi klien (browser), artinya kode dijalankan langsung oleh browser pengguna tanpa perlu dikirim kembali ke server terlebih dahulu.

Asal-Usul dan Sejarah Singkat

Dikembangkan oleh: **Brendan Eich**

Tahun lahir: 1995

Perusahaan pengembang: Netscape Communications

Nama awal: Mocha → LiveScript → JavaScript

Walau namanya mengandung kata "Java", JavaScript dan Java adalah dua bahasa yang berbeda, baik dari segi sintaks, tujuan, maupun arsitektur.

Mengapa JavaScript diciptakan?

Saat internet masih muda, halaman web bersifat statis. Netscape ingin memberikan kemampuan dinamis dan interaktif di dalam browser mereka (Netscape Navigator). Brendan Eich ditugaskan untuk membuat bahasa scripting yang dapat:

- Menyisipkan logika langsung ke halaman HTML
 - Memberi respon terhadap aksi pengguna
 - Memanipulasi tampilan tanpa harus memuat ulang halaman
-

Fungsi dan Kegunaan JavaScript

JavaScript dapat digunakan untuk:

1. Memvalidasi Form di Browser

Penjelasan:

JavaScript bisa digunakan untuk mengecek input pengguna secara real-time sebelum dikirim ke server. Ini meningkatkan kecepatan dan kenyamanan pengguna.

Contoh:

```
<form onsubmit="return validasiForm()">
```

```

<input type="email" id="email" placeholder="Email"><br>
<input type="password" id="password" placeholder="Password"><br>
<button type="submit">Kirim</button>
</form>
<p id="pesan"></p>

<script>
function validasiForm() {
    const email = document.getElementById("email").value;
    const pass = document.getElementById("password").value;
    if (email === "" || pass.length < 6) {
        document.getElementById("pesan").innerText = "Email wajib dan password minimal 6 karakter.";
        return false; // mencegah form dikirim
    }
    return true;
}
</script>

```

2. Mengubah Elemen HTML dan CSS Secara Dinamis

Penjelasan:

Dengan JavaScript, kamu bisa mengubah konten, struktur, dan tampilan HTML/CSS kapan saja tanpa memuat ulang halaman.

Contoh:

```

<p id="teks">Teks awal</p>
<button onclick="ubah()">Ubah Teks & Warna</button>

<script>
function ubah() {
    const teks = document.getElementById("teks");
    teks.innerText = "Teks sudah diubah!";
    teks.style.color = "blue";
    teks.style.fontSize = "20px";
}
</script>

```

3. Menangani Event (klik, hover, input)

Penjelasan:

JavaScript memungkinkan kita untuk merespons interaksi pengguna, seperti klik tombol, ketik input, atau arahkan mouse.

Contoh:

```

<button id="klik">Klik Aku</button>
<p id="hasil"></p>

<script>

```

```
document.getElementById("klik").addEventListener("click", function() {
    document.getElementById("hasil").innerText = "Tombol sudah diklik!";
});
</script>
```

✓ 4. Membuat Animasi, Slideshow, dan Drag & Drop

💡 Animasi (tanpa library):

```
<div id="kotak" style="width:50px; height:50px; background:red;
position:relative;"></div>
<button onclick="gerakkan()">Gerakkan</button>

<script>
function gerakkan() {
    let elemen = document.getElementById("kotak");
    let posisi = 0;
    const gerak = setInterval(() => {
        if (posisi >= 200) clearInterval(gerak);
        else {
            posisi++;
            elemen.style.left = posisi + "px";
        }
    }, 5);
}
</script>
```

💡 Slideshow sederhana:

```

<button onclick="next()">Next</button>

<script>
const gambar = ["img1.jpg", "img2.jpg", "img3.jpg"];
let i = 0;
function next() {
    i = (i + 1) % gambar.length;
    document.getElementById("slide").src = gambar[i];
}
</script>
```

✓ 5. Berkommunikasi dengan Server melalui AJAX

Penjelasan:

AJAX (Asynchronous JavaScript and XML) memungkinkan pengambilan data dari server tanpa reload halaman.

Contoh sederhana (mengambil data JSON):

```
<button onclick="ambilData()">Ambil Data</button>
```

```
<p id="output"></p>

<script>
function ambilData() {
    const xhr = new XMLHttpRequest();
    xhr.open("GET", "data.json", true); // file JSON lokal atau dari server
    xhr.onload = function() {
        if (xhr.status === 200) {
            const data = JSON.parse(xhr.responseText);
            document.getElementById("output").innerText = "Nama: " + data.nama;
        }
    };
    xhr.send();
}
</script>
```

Atau gunakan `fetch()` untuk versi modern.

✓ 6. Mengembangkan Aplikasi SPA (Single Page Application)

Penjelasan:

SPA adalah aplikasi web yang tidak me-reload seluruh halaman, melainkan hanya mengganti konten secara dinamis via JavaScript. Biasanya dibangun dengan framework seperti React, Vue, Angular, namun bisa juga dibuat dengan JavaScript murni.

Contoh sederhana SPA manual:

```
<button onclick="tampilkan('home')">Home</button>
<button onclick="tampilkan('tentang')">Tentang</button>

<div id="konten"></div>

<script>
function tampilkan(halaman) {
    if (halaman === 'home') {
        document.getElementById("konten").innerHTML = "<h2>Ini halaman HOME</h2>";
    } else if (halaman === 'tentang') {
        document.getElementById("konten").innerHTML = "<h2>Ini halaman TENTANG</h2>";
    }
}
</script>
```

✓ 7. Dipakai juga di Server (dengan Node.js)

Penjelasan:

Dengan Node.js, JavaScript bisa berjalan di luar browser, biasanya di server. Kamu bisa membuat API, CLI, web server, socket server, dll.

Contoh script Node.js (server HTTP sederhana):

```
// file: server.js
```

```

const http = require('http');

const server = http.createServer((req, res) => {
  res.writeHead(200, {"Content-Type": "text/plain"});
  res.end("Halo dari server Node.js");
});

server.listen(3000, () => {
  console.log("Server berjalan di http://localhost:3000");
});

```

Fitur	Fungsi Utama
Validasi form	Cek input sebelum kirim ke server
Manipulasi elemen	Ubah HTML & CSS secara real-time
Event handling	Tanggapi interaksi pengguna
Animasi/Slideshow	Efek visual & UX menarik tanpa reload
AJAX	Ambil data tanpa reload
SPA	Navigasi cepat tanpa reload seluruh halaman
Node.js	Gunakan JavaScript di sisi server, bukan hanya di browser

Perkembangan Modern JavaScript

Seiring waktu, JavaScript berevolusi dari bahasa scripting sederhana menjadi bahasa pemrograman penuh yang mendukung:

- OOP (Object-Oriented Programming)
 - Functional Programming
 - Modul dan ES6+ syntax (let, const, arrow function, class, import/export)
 - Framework besar seperti React, Angular, Vue.js
 - Eksekusi server-side lewat Node.js
-

Standarisasi: ECMAScript

JavaScript sekarang berada di bawah standar internasional yang disebut ECMAScript (ES), yang dikembangkan oleh ECMA International. Setiap versi ECMAScript menambahkan fitur-fitur baru, seperti:

Versi	Tahun	Fitur Penting
ES5	2009	Strict mode, JSON
ES6	2015	let/const, arrow function, class, module
ES7+	2016+	async/await, optional chaining, dll

JavaScript di Mana Saja

Dulu JavaScript hanya di browser. Sekarang dapat digunakan untuk:

- Frontend Web: Interaktivitas (HTML/CSS/JS)
 - Backend: API server dengan Node.js
 - Mobile App: React Native, Ionic
 - Desktop App: Electron (misalnya aplikasi VS Code)
 - IoT & Robotics: Johnny-Five (Arduino pakai JS)
 - Machine Learning: TensorFlow.js
-

Ciri Khas JavaScript

Karakteristik	Penjelasan
Interpreted	Tidak perlu dikompilasi, langsung dijalankan di browser
Dynamic Typing	Tipe data tidak perlu ditentukan secara eksplisit
Event-driven	Respon terhadap aksi pengguna
Prototypal Inheritance	Pewarisan menggunakan objek, bukan class murni
Versatile	Bisa di backend maupun frontend

Contoh Sederhana JavaScript

```
<!DOCTYPE html>
<html>
<head>
    <title>Contoh JavaScript</title>
</head>
<body>
    <h1 id="judul">Selamat datang!</h1>
    <button onclick="ubahJudul()">Klik Saya</button>

    <script>
        function ubahJudul() {
            document.getElementById("judul").innerText = "Halo, JavaScript aktif!";
        }
    </script>
</body>
</html>
```

Bagaimana Cara Menulis Program JavaScript?

➊ A. Internal JavaScript

📌 Penjelasan:

Internal JavaScript ditulis langsung di dalam file HTML menggunakan tag `<script>` di dalam `<head>` atau sebelum penutup `</body>`.

✓ Kelebihan:

- Praktis untuk halaman kecil atau script pendek.
- Mudah didebug karena dalam satu file.

✗ Kekurangan:

- Tidak modular (sulit digunakan ulang di halaman lain).
- Kurang rapi jika script terlalu panjang.

💡 Contoh:

```
<!DOCTYPE html>
<html>
<head>
    <title>Internal JS</title>
    <script>
        function salam() {
            alert("Selamat datang!");
        }
    </script>
</head>
<body>
    <button onclick="salam()">Klik untuk salam</button>
</body>
</html>
```

🧠 Kapan digunakan?

Cocok untuk halaman tunggal atau halaman prototipe.

➋ B. Inline JavaScript

📌 Penjelasan:

Inline JavaScript langsung ditulis dalam atribut HTML, seperti `onclick`, `onmouseover`, dll.

✓ Kelebihan:

- Sangat cepat dan mudah untuk interaksi sederhana.
- Tidak perlu deklarasi fungsi terpisah.

✗ Kekurangan:

- Tidak sesuai prinsip separasi antara konten dan logika.
- Tidak skalabel untuk proyek besar.
- Sulit dikelola jika banyak interaksi.

💡 Contoh:

```
<button onclick="alert('Anda mengklik tombol!')">Klik Saya</button>
```

💡 *Kapan digunakan?*

Cocok untuk uji coba cepat atau interaksi sederhana sekali pakai.

📍 C. Eksternal JavaScript

📌 *Penjelasan:*

Script JavaScript ditempatkan di file terpisah (.js), lalu disisipkan menggunakan tag `<script src="nama_file.js"></script>`.

✓ *Kelebihan:*

- Bersih dan terorganisir.
- Bisa digunakan ulang di banyak halaman.
- Mempercepat loading awal karena bisa di-cache oleh browser.

✗ *Kekurangan:*

- Tidak bisa langsung melihat kodennya di file HTML.
- Perlu koneksi tambahan ke file .js.

💡 *Contoh:*

File HTML:

```
<!DOCTYPE html>
<html>
<head>
    <title>Eksternal JS</title>
    <script src="fungsi.js"></script>
</head>
<body>
    <button onclick="salam()">Halo!</button>
</body>
</html>
```

File fungsi.js:

```
function salam() {
    alert("Halo dari file eksternal!");
}
```

💡 *Kapan digunakan?*

Cocok untuk aplikasi atau situs berskala besar agar kode modular dan mudah dipelihara.

🎯 Ringkasan Tabel

Cara	Kelebihan	Kekurangan	Cocok Untuk
Internal	Mudah & cepat, semua dalam 1 file	Tidak modular	Halaman kecil / demo
Inline	Praktis untuk aksi cepat	Tidak terstruktur, sulit dirawat	Interaksi sangat sederhana
Eksternal	Reusable, rapi, scalable	Perlu file terpisah	Aplikasi besar, banyak halaman

DOM (Document Object Model)

🧠 Apa itu DOM?

DOM (Document Object Model) adalah representasi struktur dokumen HTML sebagai pohon (tree). Dengan DOM, JavaScript dapat:

- Membaca konten HTML
 - Mengubah elemen atau teks
 - Menambah dan menghapus elemen
 - Merespons aksi pengguna (klik, input, dll)
-

🌲 Struktur DOM Sederhana

Contoh HTML:

```
<html>
  <body>
    <h1 id="judul">Selamat Datang</h1>
    <p class="paragraf">Ini paragraf pertama</p>
  </body>
</html>
```

Direpresentasikan oleh DOM seperti ini:

```
Document
  └── html
    └── body
      ├── h1#judul
      └── p.paragraf
```

✳️ Cara Mengakses Elemen DOM

1. By ID

```
document.getElementById("judul")
```

2. By Class

```
document.getElementsByClassName("paragraf")[0]
```

3. By Tag Name

```
document.getElementsByTagName("p")[0]
```

4. By CSS Selector

```
document.querySelector(".paragraf")
document.querySelectorAll("p")[1]
```

🔧 Memanipulasi Konten dan Atribut

1. Mengubah Teks

```
document.getElementById("judul").innerText = "Judul Baru";
```

2. Mengubah HTML

```
document.getElementById("judul").innerHTML = "<em>Judul Baru</em>";
```

3. Mengubah Atribut

```
document.getElementById("judul").setAttribute("class", "besar");
```

4. Mengubah Style Langsung

```
document.getElementById("judul").style.color = "blue";
```

Menambahkan atau Menghapus Elemen

Menambahkan Elemen

```
let pBaru = document.createElement("p");
pBaru.innerText = "Ini paragraf tambahan";
document.body.appendChild(pBaru);
```

 Penjelasan Langkah per Langkah:

1. `document.createElement("p")`

Membuat sebuah elemen HTML baru (dalam hal ini tag `<p>`), tapi elemen ini belum muncul di halaman, masih tersimpan di memori JavaScript.

2. `pBaru.innerText = "..."`

Memberikan isi teks pada elemen tersebut. Kamu juga bisa menggunakan `innerHTML` kalau ingin menyisipkan HTML.

3. `appendChild(pBaru)`

Barulah elemen ini dimasukkan ke dalam dokumen HTML — di akhir dari elemen target (`document.body` berarti di akhir `<body>`).

Menghapus Elemen

```
let elemen = document.getElementById("judul");
elemen.remove();
```

 Penjelasan Langkah per Langkah:

1. `getElementById("judul")`

Mengambil elemen HTML yang memiliki `id="judul"`.

2. `.remove()`

Fungsi ini langsung menghapus elemen dari halaman DOM. Setelah dihapus, elemen tersebut **hilang secara permanen dari halaman**, kecuali ditambahkan kembali.

Menangani Event DOM

JavaScript bisa mendeteksi aksi pengguna melalui event handler.

1. Klik

```
document.getElementById("tombol").onclick = function() {
    alert("Tombol diklik!");
};
```

2. Input

```
document.getElementById("inputku").oninput = function() {
    console.log(this.value);
};
```

Namun tidak hanya itu, berikut table lengkap mengenai event DOM:

1. Mouse Events (pergerakan dan klik mouse)

Event	Deskripsi	Contoh
onclick	Ketika elemen diklik	button.onclick = () => alert("Klik!");
ondblclick	Klik ganda	div.ondblclick = () => alert("Double click!");
onmouseover	Saat kursor masuk ke elemen	div.onmouseover = () => div.style.color = 'red';
onmouseout	Saat kursor keluar dari elemen	div.onmouseout = () => div.style.color = 'black';
onmousemove	Saat kursor bergerak di atas elemen	div.onmousemove = () => console.log("Bergerak");

2. Keyboard Events (tekanan tombol keyboard)

Event	Deskripsi	Contoh
onkeydown	Saat tombol ditekan (bisa terus saat ditekan)	document.onkeydown = e => console.log(e.key);
onkeypress	Saat tombol karakter ditekan	(sedikit usang, lebih baik pakai keydown)
onkeyup	Saat tombol dilepas	input.onkeyup = () => console.log("Naik");

3. Form Events (interaksi dengan form)

Event	Deskripsi	Contoh
oninput	Saat isi input berubah	input.oninput = () => console.log(input.value);
onchange	Saat nilai input berubah & kehilangan fokus	select.onchange = () => alert("Diubah");
onsubmit	Saat form dikirim	form.onsubmit = e => { e.preventDefault(); alert("Form disubmit"); }
onfocus	Saat elemen mendapatkan fokus	input.onfocus = () => input.style.background = 'yellow';
onblur	Saat elemen kehilangan fokus	input.onblur = () => input.style.background = '';

4. Window Events

Event	Deskripsi	Contoh
onload	Saat halaman selesai dimuat	window.onload = () => alert("Halaman siap!");

onresize	Saat ukuran jendela berubah	window.onresize = () => console.log("Resize!");
onscroll	Saat halaman di-scroll	window.onscroll = () => console.log("Scrolling...");

💡 5. Drag and Drop Events (seret dan lepas elemen)

Event	Deskripsi
ondragstart	Saat elemen mulai diseret
ondragover	Saat elemen diseret melewati area lain
ondrop	Saat elemen dijatuhkan

💡 Contoh Aplikasi DOM Sederhana

✓ Ganti Warna Paragraf

```
<p id="teks">Paragraf ini bisa diganti warnanya</p>
<button onclick="ubahWarna()">Ubah Warna</button>

<script>
    function ubahWarna() {
        const p = document.getElementById("teks");
        p.style.color = "red";
        p.style.fontWeight = "bold";
    }
</script>
```

✓ Menambah Daftar Secara Dinamis

```
<ul id="daftar"></ul>
<button onclick="tambahItem()">Tambah Item</button>

<script>
    function tambahItem() {
        let item = document.createElement("li");
        item.innerText = "Item baru";
        document.getElementById("daftar").appendChild(item);
    }
</script>
```

💡 Tips Penting dalam Manipulasi DOM

Praktik Baik	Penjelasan
Gunakan querySelector	Lebih fleksibel karena bisa gunakan selector CSS
Pisahkan logika ke JS	Jangan pakai inline JavaScript di HTML

Gunakan event listener	Lebih modular, tidak campur dengan HTML
Hindari perubahan DOM berlebihan	Bisa membuat halaman lambat jika sering dimodifikasi langsung

📌 Ringkasan

Aksi DOM	Contoh
Ambil elemen	<code>document.getElementById("id")</code>
Ganti teks	<code>elemen.innerText = "Teks baru"</code>
Ganti gaya CSS	<code>elemen.style.color = "red"</code>
Tambah elemen	<code>document.createElement("tag")</code>
Event klik	<code>elemen.onclick = function() {...}</code>
