Nama : Muhammad Zulfikran

NIM : 22650174

HTML (HyperText Markup Language)

HTML adalah bahasa markup yang digunakan untuk membuat dan menyusun struktur halaman web. HTML bukanlah bahasa pemrograman, melainkan bahasa markup yang mengatur bagaimana konten ditampilkan di browser. Dalam HTML, elemen-elemen seperti teks, gambar, tautan, dan formulir diatur dalam struktur yang mudah dipahami oleh browser.

Berikut adalah beberapa poin penting tentang HTML:

1. Struktur Dasar HTML

HTML memiliki struktur dasar yang mirip dengan dokumen, dengan elemen-elemen yang menyusun hierarki. Setiap dokumen HTML dimulai dengan elemen <html>, dan di dalamnya terdapat dua bagian utama:

- **<head>**: Bagian ini berisi informasi meta, seperti judul halaman, charset, dan link ke file CSS atau JavaScript eksternal.
- **<body>**: Di sinilah semua konten utama berada, termasuk teks, gambar, tautan, video, dan elemen-elemen lain yang tampil di halaman web.

Berikut contoh struktur HTML dasar:

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Judul Halaman</title>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h1>Selamat Datang di Situs Kami</h1>
    <p>Ini adalah contoh paragraf dalam HTML.</p>
  </body>
</html>
```

2. Elemen HTML

HTML terdiri dari berbagai elemen yang masing-masing memiliki fungsi tertentu. Elemen HTML diapit oleh tag pembuka (<tag>) dan tag penutup (</tag>). Contoh elemen HTML termasuk:

- **Heading**: <h1> hingga <h6> untuk judul dengan ukuran berbeda.
- **Paragraf**: <p> untuk membuat paragraf teks.
- **Gambar**: <img src="url_gambar" alt="deskripsi"> untuk menampilkan gambar.
- **Tautan**: Teks Tautan untuk membuat link ke halaman lain atau bagian lain dalam halaman yang sama.

3. Atribut HTML

Atribut adalah informasi tambahan yang ditempatkan dalam tag untuk memberikan spesifikasi lebih lanjut pada elemen HTML. Misalnya:

- **src** pada tag <img> untuk menentukan URL gambar.
- **href** pada tag <a> untuk menentukan URL tautan.
- **class** dan **id** untuk memberikan nama pada elemen yang bisa digunakan oleh CSS atau JavaScript.

Contoh penggunaan atribut:

```html
<img src="gambar.jpg" alt="Contoh Gambar" width="200" height="150">
<a href="https://contoh.com" target="_blank">Kunjungi Contoh.com</a>
```

4. Formulir dalam HTML

HTML juga memungkinkan pembuatan formulir yang dapat mengumpulkan data dari pengguna, seperti nama, email, dan pesan. Formulir dibuat dengan elemen <form>, yang di dalamnya bisa berisi berbagai elemen input:

- **Input Teks**: <input type="text">
- **Area Teks**: <textarea></textarea>
- **Tombol Kirim**: <button type="submit">Kirim</button>

Contoh:

```html
<form action="submit_form.php" method="POST">
  <label for="nama">Nama:</label>
  <input type="text" id="nama" name="nama">
  <label for="email">Email:</label>
  <input type="email" id="email" name="email">
  <button type="submit">Kirim</button>
</form>
```

5. HyperText dan Navigasi

HTML memungkinkan pembuatan tautan (link) yang dapat membawa pengguna dari satu halaman ke halaman lain atau dari satu bagian halaman ke bagian lain (anchor). Ini adalah fitur utama HTML yang membuatnya disebut sebagai **HyperText**.

Contoh tautan:

```html
<a href="halaman-lain.html">Pergi ke Halaman Lain</a>
```

6. Multimedia di HTML

HTML mendukung berbagai elemen untuk menyematkan multimedia, seperti audio, video, dan gambar:

- **Audio**: <audio src="audio.mp3" controls></audio>
- **Video**: <video src="video.mp4" controls></video>
- **Gambar**: <img src="gambar.jpg" alt="Contoh Gambar">

7. CSS dan HTML

HTML digunakan bersama dengan CSS (Cascading Style Sheets) untuk mempercantik tampilan web. CSS memungkinkan pengaturan warna, font, layout, dan animasi, sementara HTML berfungsi untuk struktur konten.

8. JavaScript dan HTML

JavaScript adalah bahasa pemrograman yang digunakan bersama HTML untuk menambahkan interaktivitas pada halaman web. JavaScript bisa ditempatkan langsung di HTML menggunakan tag <script>, atau di-link sebagai file eksternal.

9. HTML Responsif

HTML sekarang mendukung responsivitas, yang artinya halaman web dapat menyesuaikan tampilannya berdasarkan perangkat yang digunakan (desktop, tablet, atau ponsel). Responsivitas sering dicapai dengan bantuan CSS media query dan elemen HTML5 yang lebih fleksibel.

10. HTML5

HTML5 adalah versi terbaru HTML yang menambahkan berbagai fitur baru, seperti elemen struktur (<header>, <footer>, <article>, <section>), form API yang lebih canggih, serta dukungan multimedia lebih baik tanpa perlu plugin eksternal.

Contoh Lengkap Dokumen HTML5

Berikut adalah contoh sederhana dokumen HTML5:

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Halaman HTML5 Contoh</title>
  </head>
  <body>
    <header>
      <h1>Selamat Datang di Website Kami</h1>
    </header>
    <section>
      <h2>Tentang Kami</h2>
      <p>Ini adalah contoh situs web sederhana menggunakan HTML5.</p>
    </section>
    <footer>
      <p>&copy; 2024 Contoh Inc.</p>
    </footer>
  </body>
</html>
```

DEKLARASI VARIABEL JAVASCRIPTS

Dalam JavaScript, variabel adalah tempat penyimpanan data yang bisa digunakan dan dimanipulasi dalam kode. Variabel dideklarasikan menggunakan kata kunci khusus, yaitu var, let, dan const. Ketiga kata kunci ini memiliki perbedaan penting yang memengaruhi cara variabel berperilaku, terutama dalam hal lingkup (scope) dan kemampuan untuk diubah.

Berikut adalah penjelasan lengkap mengenai deklarasi variabel di JavaScript:

1. Deklarasi Variabel dengan var

var adalah cara yang lebih lama untuk mendeklarasikan variabel di JavaScript dan memiliki beberapa karakteristik khusus:

- **Lingkup Fungsi** (Function Scope): Variabel yang dideklarasikan dengan var hanya akan tersedia di dalam fungsi tempat mereka dideklarasikan, jika berada di dalam fungsi. Di luar fungsi, var bersifat global.
- **Dapat Diubah (Mutable)**: Variabel yang dideklarasikan dengan var dapat diubah nilainya kapan saja.
- **Hoisting**: Variabel yang dideklarasikan dengan var diangkat (hoisted) ke atas lingkup fungsinya, yang berarti mereka dideklarasikan di bagian atas lingkup tersebut walaupun sebenarnya ditulis di bagian bawah. Namun, nilai variabelnya baru ditetapkan saat dieksekusi.

Contoh:

```javascript
var nama = "John";
console.log(nama); // Output: John

var nama = "Doe";  // Re-deklarasi dengan var diizinkan
console.log(nama); // Output: Doe
```

Pada kode di atas, variabel nama bisa dideklarasikan ulang dan nilainya dapat diubah.

2. Deklarasi Variabel dengan let

let adalah cara baru yang diperkenalkan dalam ES6 (ECMAScript 2015) untuk mendeklarasikan variabel. let memiliki beberapa perbedaan penting dibandingkan dengan var:

- **Lingkup Blok** (Block Scope): Variabel yang dideklarasikan dengan let hanya tersedia di dalam blok tempat mereka dideklarasikan (yaitu dalam kurung {}), seperti dalam if, for, atau fungsi.
- **Dapat Diubah (Mutable)**: Nilai dari variabel yang dideklarasikan dengan let dapat diubah, tetapi tidak dapat dideklarasikan ulang dalam lingkup yang sama.
- **Hoisting Terbatas**: Meskipun let di-hoist, variabel ini tidak bisa diakses sebelum deklarasinya dalam blok yang sama. Ini dikenal sebagai "temporal dead zone".

Contoh:

javascript Copy code

```javascript
let umur = 25;
console.log(umur); // Output: 25

umur = 30;         // Nilai variabel bisa diubah
console.log(umur); // Output: 30

// let umur = 40;  // Error: variabel "umur" tidak bisa dideklarasikan ulang dalam blok ya
```

Jika let digunakan di dalam blok, variabel tersebut hanya bisa diakses dalam blok tersebut:

javascript Copy code

```javascript
if (true) {
  let warna = "merah";
  console.log(warna); // Output: merah
}
// console.log(warna); // Error: warna tidak terdefinisi di luar blok
```

3. Deklarasi Variabel dengan const

const adalah cara mendeklarasikan variabel yang nilainya **tidak dapat diubah** (konstan). const juga diperkenalkan dalam ES6 dan memiliki karakteristik berikut:

- **Lingkup Blok** (Block Scope): Seperti let, variabel yang dideklarasikan dengan const hanya tersedia di dalam blok tempat mereka dideklarasikan.
- **Tidak Dapat Diubah (Immutable)**: Nilai variabel const tidak dapat diubah setelah dideklarasikan. Jika variabel ini berupa tipe data referensi seperti objek atau array, nilai properti atau elemen di dalamnya masih bisa diubah.
- **Harus Diberikan Nilai Awal**: Variabel const harus diberikan nilai saat dideklarasikan, dan nilai tersebut tidak bisa diubah setelahnya.

Contoh:

```javascript
const pi = 3.14;
console.log(pi); // Output: 3.14

// pi = 3.1415;  // Error: nilai variabel const tidak bisa diubah
```

Namun, jika const digunakan untuk objek atau array, nilai dalam objek atau array itu bisa diubah, tetapi referensi utama ke objek atau array tidak bisa diubah:

```javascript
const warna = ["merah", "biru", "hijau"];
warna.push("kuning");
console.log(warna); // Output: ["merah", "biru", "hijau", "kuning"]

// warna = ["ungu"]; // Error: referensi ke array tidak bisa diubah
```

4. Perbedaan Utama antara var, let, dan const

Perbedaan	var	let	const
Lingkup (Scope)	Fungsi (Function Scope)	Blok (Block Scope)	Blok (Block Scope)
Dapat Diubah	Ya	Ya	Tidak
Hoisting	Ya, tetapi nilainya undefined saat diakses sebelum deklarasi	Ya, tetapi tidak bisa diakses sebelum deklarasi	Ya, tetapi tidak bisa diakses sebelum deklarasi
Deklarasi Ulang	Ya	Tidak (dalam lingkup yang sama)	Tidak
Inisialisasi Awal	Tidak wajib	Tidak wajib	Wajib

Kesimpulan

- Gunakan let saat membutuhkan variabel yang nilainya akan diubah dan hanya ingin lingkup variabel terbatas pada blok.
- Gunakan const untuk variabel yang nilainya konstan dan tidak akan diubah setelah deklarasi.
- var jarang digunakan dalam JavaScript modern karena let dan const memberikan lingkup yang lebih aman.

DOM (Document Object Model)

DOM adalah antarmuka pemrograman yang merepresentasikan struktur halaman web. Dalam DOM, setiap elemen HTML pada halaman web dipetakan sebagai objek sehingga dapat diakses dan dimodifikasi dengan JavaScript. DOM memungkinkan JavaScript untuk berinteraksi dengan elemen-elemen HTML secara dinamis, seperti mengambil nilai dari elemen atau mengubah konten, gaya, dan atribut elemen HTML.

Berikut adalah penjelasan tentang bagaimana JavaScript berinteraksi dengan HTML melalui DOM, terutama dalam hal **mengambil nilai dari elemen HTML** dan **memodifikasi konten HTML**.

1. Mengambil Nilai dari HTML Menggunakan JavaScript

JavaScript dapat mengambil nilai atau konten dari elemen HTML menggunakan beberapa metode DOM. Ada beberapa cara untuk memilih elemen dalam DOM:

- **getElementById**: Mengambil elemen berdasarkan atribut id.
- **getElementsByClassName**: Mengambil elemen-elemen berdasarkan nama kelas (class name).
- **getElementsByTagName**: Mengambil elemen-elemen berdasarkan nama tag (tag name).
- **querySelector**: Mengambil elemen pertama yang cocok dengan selector CSS.
- **querySelectorAll**: Mengambil semua elemen yang cocok dengan selector CSS.

Setelah elemen diambil, kita bisa mengakses nilai atau konten elemen tersebut menggunakan properti seperti .value (untuk elemen input), atau .innerText dan .innerHTML (untuk teks dan konten HTML lainnya).

Contoh 1: Mengambil Nilai dari Input dengan getElementById

html Copy code

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Contoh Mengambil Nilai dengan JavaScript</title>
</head>
<body>
  <input type="text" id="namaInput" placeholder="Masukkan nama Anda">
  <button onclick="ambilNilai()">Ambil Nilai</button>
  <p id="output"></p>

  <script>
    function ambilNilai() {
      // Mengambil elemen input berdasarkan id
      const inputElement = document.getElementById("namaInput");
      const nama = inputElement.value; // Mengambil nilai dari input

      // Menampilkan hasil
      document.getElementById("output").innerText = "Nama Anda: " + nama;
    }
  </script>
</body>
</html>
```

Pada contoh di atas:

- **document.getElementById("namaInput")** mengambil elemen input berdasarkan id.
- **.value** digunakan untuk mendapatkan nilai yang diinputkan pengguna.
- Hasil kemudian ditampilkan di paragraf dengan id="output".

Contoh 2: Mengambil Teks dari Paragraf dengan querySelector

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Contoh Mengambil Teks</title>
</head>
<body>
  <p class="paragraf">Ini adalah teks contoh.</p>
  <button onclick="ambilTeks()">Ambil Teks</button>
  <p id="hasilTeks"></p>

  <script>
    function ambilTeks() {
      // Mengambil elemen paragraf berdasarkan kelas (class)
      const paragrafElement = document.querySelector(".paragraf");
      const teks = paragrafElement.innerText; // Mengambil teks dari elemen

      // Menampilkan hasil
      document.getElementById("hasilTeks").innerText = "Teks yang diambil: " + teks;
    }
  </script>
</body>
</html>
```

Pada contoh ini, querySelector memilih elemen pertama dengan kelas .paragraf dan mengambil teksnya menggunakan .innerText.

2. Memodifikasi HTML Menggunakan JavaScript

JavaScript memungkinkan kita untuk memodifikasi elemen HTML di DOM, termasuk mengubah teks, atribut, gaya CSS, atau menambahkan/menghapus elemen.

Beberapa metode umum untuk memodifikasi elemen di HTML:

- **Mengubah Teks atau Konten**: Menggunakan .innerText atau .innerHTML.
- **Mengubah Atribut**: Menggunakan .setAttribute atau properti atribut tertentu (seperti .src untuk gambar).

- **Mengubah Gaya CSS**: Menggunakan .style untuk menambahkan atau mengubah gaya CSS elemen.
- **Menambah atau Menghapus Elemen**: Menggunakan appendChild, removeChild, atau innerHTML.

Contoh 1: Mengubah Teks atau Konten Elemen

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Contoh Mengubah Teks</title>
</head>
<body>
  <p id="paragraf">Teks awal.</p>
  <button onclick="ubahTeks()">Ubah Teks</button>

  <script>
    function ubahTeks() {
      // Mengubah teks elemen dengan id "paragraf"
      document.getElementById("paragraf").innerText = "Teks telah diubah!";
    }
  </script>
</body>
</html>
```

Pada contoh ini:

- **document.getElementById("paragraf").innerText** digunakan untuk mengubah teks dalam elemen <p>.

Contoh 2: Mengubah Atribut Gambar

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Contoh Mengubah Atribut</title>
</head>
<body>
  <img id="gambar" src="gambar1.jpg" alt="Gambar Awal" width="200">
  <button onclick="ubahGambar()">Ubah Gambar</button>

  <script>
    function ubahGambar() {
      // Mengubah atribut src pada elemen gambar
      document.getElementById("gambar").src = "gambar2.jpg";
    }
  </script>
</body>
</html>
```

Pada contoh ini, **.src** digunakan untuk mengubah atribut src pada elemen gambar.

Contoh 3: Mengubah Gaya CSS

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Contoh Mengubah Gaya</title>
</head>
<body>
  <p id="paragrafGaya">Teks dengan gaya yang bisa diubah.</p>
  <button onclick="ubahGaya()">Ubah Gaya</button>

  <script>
    function ubahGaya() {
      // Mengubah gaya CSS dari elemen paragraf
      const paragraf = document.getElementById("paragrafGaya");
      paragraf.style.color = "blue";        // Mengubah warna teks
      paragraf.style.fontSize = "24px";     // Mengubah ukuran font
      paragraf.style.fontWeight = "bold";   // Mengubah ketebalan teks
    }
  </script>
</body>
</html>
```

Pada contoh ini:

- **.style.property** digunakan untuk mengatur properti CSS elemen, seperti color, fontSize, dan fontWeight.

Contoh 4: Menambah Elemen Baru

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Contoh Menambah Elemen</title>
</head>
<body>
  <div id="kontainer">
    <p>Elemen awal dalam kontainer</p>
  </div>
  <button onclick="tambahElemen()">Tambah Elemen</button>

  <script>
    function tambahElemen() {
      // Membuat elemen paragraf baru
      const paragrafBaru = document.createElement("p");
      paragrafBaru.innerText = "Paragraf baru yang ditambahkan.";

      // Menambahkan paragraf baru ke dalam elemen dengan id "kontainer"
      document.getElementById("kontainer").appendChild(paragrafBaru);
    }
  </script>
</body>
</html>
```

Pada contoh ini:

- **document.createElement("p")** membuat elemen <p> baru.
- **appendChild** menambahkan elemen baru ke dalam elemen div dengan id="kontainer".

Kesimpulan

DOM adalah alat penting dalam JavaScript yang memungkinkan kita berinteraksi dengan halaman HTML secara dinamis. Dengan DOM, kita dapat mengambil nilai elemen HTML, mengubah konten dan atribut elemen, menerapkan gaya, dan bahkan menambahkan atau menghapus elemen di halaman.