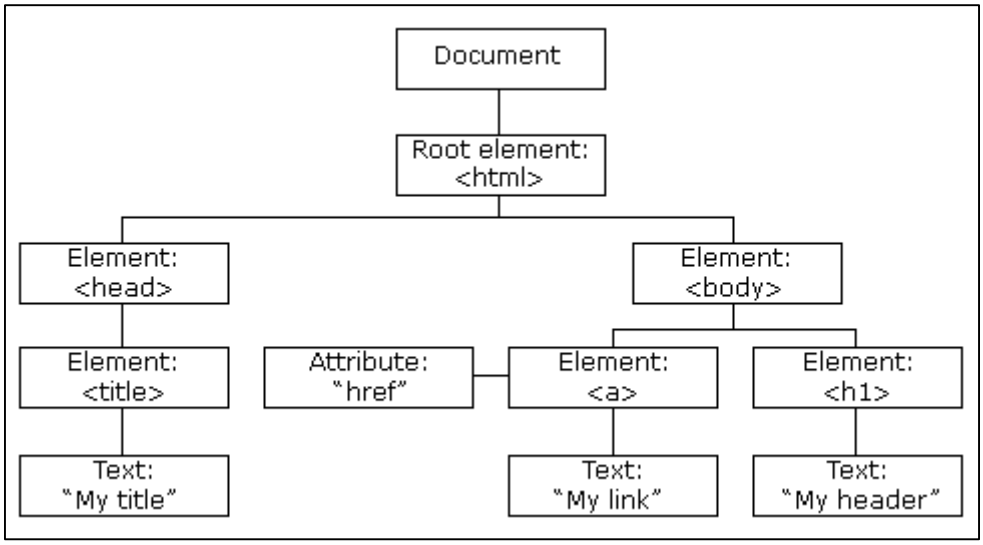
Document Object Model (DOM)

Saat halaman web dimuat, browser membuat sebuah dokumen halaman tersebut disebut dengan DOM. Fungsi DOM adalah untuk mengakses seluruh elemen-elemen halaman web, misalnya mengakses elemen <H1> , <p>, <body>. Kegunaan mengakses elemen biasanya untuk mendapatkan value-nya atau mengganti value dari elemen, Misalnya mengganti warna tulisan pada elemen <H1>.

Model HTML DOM dibangun sebagai pohon Objek:



DEFINISI

DOM menentukan standar untuk mengakses dokumen. DOM W3C adalah platform dan antarmuka tanpa bahasa yang memungkinkan program dan skrip untuk mengakses dan memperbarui konten, struktur, dan gaya dokumen secara dinamis.

Standar DOM W3C dipisahkan menjadi 3 bagian berbeda:

a. Core DOM - model standar untuk semua jenis dokumen
b. XML DOM - model standar untuk dokumen XML
c. HTML DOM - model standar untuk dokumen HTML

HTML DOM

adalah standar objek Model dan pemrograman antarmuka untuk HTML. Ini mendefinisikan :

- HTML elements sebagai objek
- Properti dari semua element HTML
- Method yang dapat mengakses semua element HTML
- Event untuk semua element HTML

Dalam kata lain, DOM adalah standar untuk melakukan bagaimana mendapatkan, merubah, menambah, dan menghapus HTML.

Cara Mendapatkan Elemen HTML

1. **Mendapatkan elemen dengan ID**

   Method getElementById() digunakan untuk mendapatkan elemen tunggal dengan id-nya. Mari kita lihat sebuah contoh:

   ```
   var title = document.getElementById('header-title');
   ```

   Di sini kita mendapatkan elemen dengan id *header-title* dan menyimpannya ke dalam variabel.

2. **Mendapatkan elemen dengan nama kelas**

   Kita juga bisa mendapatkan lebih dari satu objek dengan menggunakan *method* getElementsByClassName().

   ```
   var items = document.getElementsByClassName('list-items');
   ```

   Di sini kita mendapatkan semua item dengan kelas *list-items* dan menyimpannya ke dalam variabel.

3. **Mendapatkan elemen dengan nama *tag***

   Kita juga bisa mendapatkan elemen kita dengan nama tag menggunakan *method* getElementsByTagName().

   ```
   var listItems = document.getElementsByTagName('li');
   ```

   Di sini kita mendapatkan semua elemen *li* dari dokumen HTML kita dan menyimpannya ke dalam variabel.

Mengubah Elemen HTML

HTML DOM memungkinkan kita mengubah konten dan *style* elemen HTML dengan mengubah propertinya.

1. **Mengubah HTML**

   Properti innerHTML dapat digunakan untuk mengubah konten elemen HTML.

   ```
   document.getElementById("#header").innerHTML = "Hello World!";
   ```

   Dalam contoh ini kita mendapatkan elemen dengan id *header* dan mengatur konten *inner* menjadi "Hello World!".

   InnerHTML juga dapat digunakan untuk menempatkan *tag* di *tag* lain.

   ```
   document.getElementsByTagName("div").innerHTML = "<h1>Hello World!</h1>"
   ```

   Di sini kita meletakkan *tag* h1 ke semua div yang sudah ada.

2. **Mengubah nilai atribut**

   Anda juga dapat mengubah nilai atribut menggunakan DOM.

   ```
   document.getElementsByTag("img").src = "test.jpg";
   ```

   Dalam contoh ini kita mengubah *src* dari semua *tag* <img /> menjadi test.jpg.

3. **Mengubah *style***

   Untuk mengubah *style* elemen HTML, kita perlu mengubah properti *style* elemen kita.
   Berikut ini contoh sintaks untuk mengubah *style*:

   ```
   document.getElementById(id).style.property = new style
   ```

   Sekarang mari kita lihat contoh di mana kita mendapatkan elemen dan mengubah batas bawah menjadi garis hitam solid:

   ```
   document.getElementsByTag("h1").style.borderBottom = "solid 3px #000";
   ```

   Properti CSS perlu ditulis dalam **camelcase** bukan nama properti css normal. Dalam contoh ini kita menggunakan borderBottom.

Menambah dan menghapus elemen

Sekarang kita akan melihat bagaimana kita dapat menambahkan elemen baru dan menghapus yang sudah ada.

1. **Menambahkan elemen**

   Berikut ini contoh sintaks untuk menambah elemen:

   ```
   var div = document.createElement('div');
   ```

   Di sini kita hanya membuat elemen div menggunakan *method* createElement() yang mengambil *tagname* sebagai parameter dan menyimpannya ke dalam variabel. Setelah itu kita hanya perlu memberikan beberapa konten dan kemudian memasukkannya ke dokumen DOM kita.

   ```
   var newContent = document.createTextNode("Hello World!");
   div.appendChild(newContent);
   document.body.insertBefore(div, currentDiv);
   ```

   Di sini kita membuat konten menggunakan *method* createTextNode() yang menggunakan sebuah String sebagai parameter dan kemudian kita memasukkan elemen div baru sebelum div yang sudah ada dalam dokumen kita.

2. **Menghapus elemen**

   Berikut ini contoh sintaks untuk menghapus elemen:
   var elem = document.querySelector('#header');
   elem.parentNode.removeChild(elem);
   Di sini kita mendapatkan elemen dan menghapusnya menggunakan *method* removeChild().

3. **Mengganti elemen**

   Berikut ini contoh sintaks untuk mengganti elemen:

```
var div = document.querySelector('#div');

var newDiv = document.createElement(‘div’);
newDiv.innerHTML = "Hello World2"

div.parentNode.replaceChild(newDiv, div);
```

   Di sini kita dapat mengganti elemen menggunakan *method* replaceChild(). Argumen pertama adalah elemen baru dan argumen kedua adalah elemen yang ingin kita ganti.

Referensi:

1. https://www.w3schools.com/js/js_htmldom.asp
2. Tanner, Gabriel. 2019. An introduction to the JavaScript DOM. [Online] Available at :https://medium.freecodecamp.org/an-introduction-to-the-javascript-dom-512463dd62ec