FRONT-END

JAVASCRIPT

TEMEL DOM İŞLEMLERİ VE OLAYLAR

DOM (DOCUMENT OBJECT MODEL)

- **DOM (Belge Nesne Modeli)** dokümanların yapısallaştırılmış gösterimidir ve **W3C** (World Wide Web Consortium) standardıdır.

- DOM, dokümanlarının **stillerinin**, **yapısının, içeriğinin** erişilmesine ve güncellenmesine olanak sağlayan, dilden-bağımsız bir arabirimdir.

- **W3C DOM** standardı **3 farklı** alt birime ayrılmaktadır.
  - **Core DOM** – Bütün doküman tipleri için standart modeldir.
  - **XML DOM** – XML dokümanları için standart modeldir.
  - **HTML DOM – HTML dokümanları için standart modeldir.**

- **How a Browser Works**
  - https://www.youtube.com/watch?v=z0HN-fG6oT4

HTML DOM => KISACA DOM

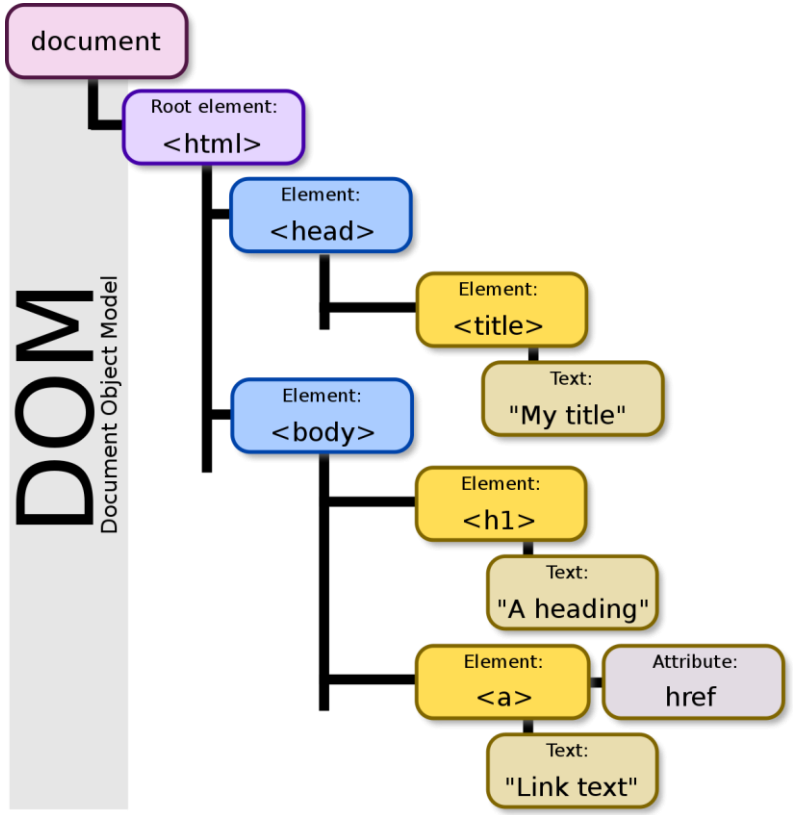
- **HTML** dokümanları için standart **nesne modeli** ve programlama arabirimidir (API).

- **DOM** şunları tanımlar;
  - HTML elemanlarını Nesneler (**objects**) olarak,
  - Tüm HTML elemanlarını özelliklerini (**properties**)
  - Tüm HTML elemanlarının erişimine olanak sağlayan metotları (**methods**),
  - Tüm HTML elemanları için olayları (**events**)

- Diğer bir ifade ile HTML elemanlarının **okunması**, **değiştirilmesi**, **eklenmesi** ve **silinmesi** gibi işlemlerin nasıl yapılacağını belirleyen bir standarttır.

DOM

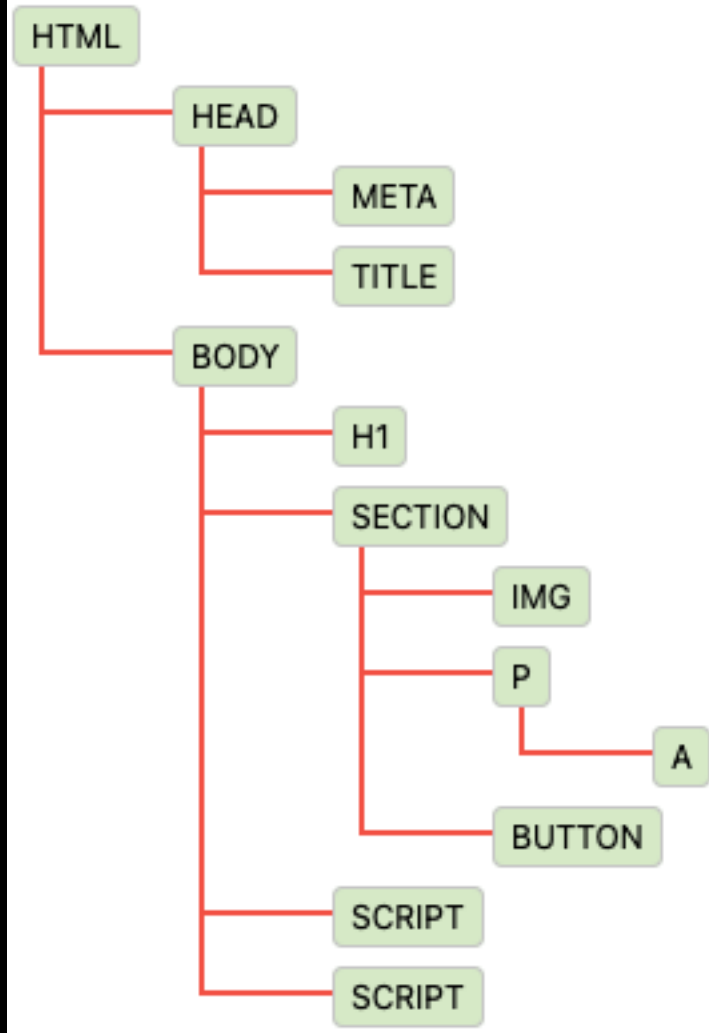
- Her tarayıcının bir **DOM API'si** bulunmaktadır. Bu API, DOM'daki elemanlara erişim için hazır bir çok metot barındırmaktadır.
- Bir web sayfası yüklendiğinde tarayıcı, ilgili HTML sayfasının **DOM'unu** oluşturur.
- DOM ağaç yapısı ise HTML dosyası içerisinde bulunan **tüm nesnelerin** hiyerarşik bir biçimde gösterimidir.
- Ağacın kökünde **<html>** elamanı **(root node)** vardır.
- **<html>** elemanın iki adet çocuk elemanı (**child node**) vardır.
  - **<head>** ve **<body>**



DOM (ÖRNEK)

DOM ağacı (HTML Tree Genarator)

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>DOM</title>
  </head>
  <body>
    <h1>DOM-INTRO</h1>
    <section>
      <img src="img/js_logo.png" alt="Javascript" />
      <p>
        Link for learning Javascript
        <a href="https://clarusway.com/" target="_blank">Javascript</a>
      </p>
      <button>Click</button>
    </section>
    <script src="selectors.js"></script>
  </body>
</html>
```



DOM ELEMANLARINA NASIL ERİŞİLİR

- Bir HTML sayfasındaki elemanları seçmek için DOM API'de bir çok hazır metot bulunmaktadır.

HTML Elemanları Seçmek için metotlar

Metot	Açıklaması
document.getElementById(id)	Bir elemanı id'ye göre bul
document.getElementsByTagName(isim)	Bir elemanı Tag (Etiket) adına göre bul
document.getElementsByClassName(isim)	Bir elemanı Class (Sınıf) adına göre bul

DOM ELEMANLARINA NASIL ERİŞİLİR

HTML Elemanları Seçmek için metotlar

Metot	Açıklaması
document.querySelector(CSS seçici)	Bir elemanı id, class, özellik, tür ve değere göre seçmek için kullanılır. Eşleşen ilk elemanı seçer.
document.querySelectorAll(CSS seçici)	Bir elemanı id, class, özellik, tür ve değere göre seçmek için kullanılır. Eşleşen elemanların listesini döndürür.

DOM ELEMANLARINA NASIL ERİŞİLİR

- HTML elemanları bir önceki slaytta bulunan metotlar ile seçildikten sonra O elemana ait bazı **özellikler** ve **stiller** değiştirilebilir.

HTML Elemanları Özellik ve Stillerini Değiştirmek

Metot	Açıklaması
element.innerHTML = yeni içerik	Bir HTML elemanın içerğini değiştirme
element.attribute = yeni değer	Bir HTML elemanının özelliğini değiştirme
element.setAttribute(özellik, değer)	Bir HTML elemanının özelliğini değiştirme
element.style.property = yeni stil	Bir HTML elemanın stilini değiştirme

YENİ BİR HTML ELEMANI NASIL OLUŞTURULUR?

Yeni bir HTML Elemanı oluşturmak için metotlar

Metot	Açıklaması
document.createElement(isim)	Yeni bir eleman oluşturur.
Document.createTextNode(text)	Bir eleman için text düğümü oluşturur.
element.appendChild(text düğüm)	Bir text düğümüne elemana bağlar.
Element.removeChild(child düğüm)	Bir elemanın çocuk düğümünü siler.

DOM ÖRNEK-1 (GETELEMENTBYID)

- Önceki Slaytlarda gösterdiğimiz HTML sayfasının bazı özellik ve stillerini Javascript ile değiştirelim.

- Sayfamızda bulunan **<p>** elemanına erişip renginde değişiklik yapalım.
  - Bunun için HTML dosyamızda p elemanına bir **id** ekleyelim

```
<p id="my-par">Javascript öğrenmek için link </p>
```

  - Artık **getElementById()** metodu ile javacript kodumuzda **<p>** elemanına erişebiliriz.

```
const paragraf = document.getElementById('my-par');
```

  - Ve artık bu elemana ait bir çok özelliğe **değişkenAdı.** İle erişebiliriz.

```
paragraf.style.color = 'red';
```

DOM ÖRNEK-2 (GETELEMENTBYID)

- Sayfamızda bulunan **<button>** elemanına ait bazı değişiklikleri yapalım.
  - Bunun için HTML dosyamızda elemanımıza **id='btn'** eklemesini yapalım.

```
const buton = document.getElementById("btn");
buton.style.color = "yellow";
buton.style.backgroundColor = "black";
buton.innerHTML = "ARA";
buton.style.fontSize = 20;
```

NOT: **id** tüm HTML dosyası için **tekrarsız** olduğu için **getElementById()** metodu ile sadece tek bir HTML elemanını seçebiliriz.

DOM ÖRNEK-3 (GETELEMENTBYTAGNAME)

- Sayfamızda bulunan **<img** elemanına ait bazı değişiklikleri yapalım.
  - Bu sefer **getElementByTagName()** metodu ile elemanımızın özelliklerine erişelim.

```
const resim = document.getElementsByTagName("img");
resim[0].style.borderStyle = "solid";
resim[0].style.borderolor = "purple";
```

NOT: Bu metot ile etiketi aynı olan tüm elemanlar **(<img)** seçilmektedir. Dolayısıyla tek bir eleman seçmek için **indisleme** kullanmak gerekir.

DOM ÖRNEK-4 (GETELEMENTBYCLASSNAME)

- Sayfamızda bulunan **<h1>** elemanına ait bazı değişiklikleri yapalım.
  - Bu sefer **getElementByClassName()** metodu ile elemanımızın özelliklerine erişelim.

```
const header = document.getElementsByClassName('header');
header[0].innerHTML = 'DOM Manipülasyonu Çok ilginç';
header[0].style.fontSize = 25;
header[0].style.fontFamily ='Tahoma';
header[0].style.color = 'blue';
```

NOT: Bu metot ile **class** (sınıfı) aynı olan tüm elemanlar **(<h1>)** seçilmektedir. Dolayısıyla tek bir eleman seçmek için **indisleme** kullanmak gerekir.

DOM ÖRNEK-5 (QUERYSELECTOR)

- Sayfamızın arka planının rengini ve sayfamızın başlığını (**title**) değiştirelim ve elemanları seçmek için **querySelector()** metodunu kullanalım.

```
<body id="body">
```

```
<title class="title">DOM</title>
```

⇨ **.html**

```
const body = document.querySelector('#body');
body.style.backgroundImage = 'linear-gradient(to right, green, yellow)';

const title = document.querySelector(".title");
title.innerHTML = "DOM Örnekleri 👀";
```

NOT:

- **querySelector** metodunda **id** ile seçim yaparken **`#id'**, **class** için ise **`.classAdı'** yazılmalıdır.
- querySelector belirtilen şarta uygun ilk elemanı seçer. Aynı class'daki birden fazla elemanı seçmek için **querySelectorAll()** metodu kullanılmalıdır.

DOM OLAYLARI (EVENTS)

- DOM sayesinde JavaScript, **HTML olayları** ile etkileşimi halinde olabilir.
- Örneğin, kullanıcı bir HTML elemanına tıkladığında (bir olay meydana geldiğinde) bir Javascript kodu çalıştırılabilir.
- Bir **olay** meydana geldiğinde bir JS kodu çalıştırmak için ilgili elemanın **olay** özelliği kullanılır.
- HTML olayları için bir çok örnek verilebilir.
  - Kullanıcı Fareyi tıkladığında (**onclick**)
  - Bir web sayfası yüklendiğinde (**onload**)
  - Fare bir elemanın üzerine geldiğinde (**onmouseover**)
  - Fare bir elemanın üzerinden ayrıldığında (**onmouseout**)
  - Bir elemanın içeriği değiştiğinde (**onchange**)
  - Bir HTML formu gönderildiğinde (**onsubmit**)
  - Klavyeden bir tuşa basıldığında (**onkeyup**)

DOM OLAYLARI (ÖRNEK-1)

<h1> elemanının üzerine **Fare ile geldiğimizde** yazı rengi **mavi**, **Fare uzaklaştığında** yeniden eski rengi olan **siyaha** çevirecek kodu yazalım.

1.YÖNTEM: HTML içerisinde olayları tanımlayarak ve kullanarak (**inline**)

```
<h1 class="header" onmouseover="style.color='black'"
onmouseout="style.color='blue'"">DOM Temelleri</h1>
```

2.YÖNTEM: HTML içerisinde olay ve fonksiyonları çağırarak ve Javascript içerisinde bunları tanımlayarak

```
<h1 class="header" onmouseover=yazıSiyah()
onmouseout=yazıMavi()>DOM Temelleri</h1>
```

⇨ **.html**

```
const header = document.querySelector(".header");
const yazıMavi = () => header.style.color = "blue";
const yazıSiyah = () => header.style.color = "black";
```

⇨ **.js**

DOM OLAYLARI (ÖRNEK-1)

3.YÖNTEM: HTML sayfasında herhangi **bir ekleme yapmaksızın sadece** Javascript ile Olayları kontrol ederek.

```
document.querySelector(".header").onmouseover = function () {
    document.querySelector(".header").style.color = "blue";
};
document.querySelector(".header").onmouseout = function () {
    document.querySelector(".header").style.color = "black";
};
```

.js

DOM OLAYLARI (ÖRNEK-1)

4.YÖNTEM: **addEventListener()** metodu ile Javascript kodunda **Olay** tanımlayarak.

Syntax: *element*.addEventListener(*olay, fonksiyon*)

```
document.querySelector(".header").addEventListener('mouseover', function() {
    document.querySelector(".header").style.color = "blue";
});

document.querySelector(".header").addEventListener("mouseout", () => {
    document.querySelector(".header").style.color = "black";
});
```

NOT: Fonksiyon tanımlaması **arrow** fonksiyon yerine **function ()** deyimi ile de yapılabilir. (Dikkat: Olay isimlendirmesi biraz farklı).

DOM OLAYLARI (ÖRNEK-2)

- Butonumuza tıklandığında web sayfasının arkaplan renginin **Yeşil** olmasını ve Butonun içindeki yazının **BAS** olarak değiştirilmesini sağlayan kodu HTML sayfasına bir ekleme yapmaksızın yazalım.

1.YÖNTEM:

```
document.getElementById("btn").onclick = function () {
    document.getElementById("body").style.backgroundColor = "green";
    document.getElementById("btn").innerHTML = "BAS";
};
```

2.YÖNTEM:

```
document.querySelector("#btn").addEventListener('click', () => {
    document.querySelector("#body").style.backgroundColor = "green";
    document.querySelector("#btn").innerHTML = "BAS";
});
```

DOM OLAYLARI (ÖRNEK-3):RENK DEĞİŞTİRİCİ

- Resimde görüldüğü gibi butona her basıldığında sayfanın arka plan rengini rasgele olarak değiştiren uygulamayı yazınız.
- **NOT:** Renkler (Mavi, Yeşil, Sarı, Siyah, Pembe v.b) dizide saklanacak Ve aktif olan renk h1 elamanı olarak ekranda yazdırılacaktır.

DOM OLAYLARI (ÖRNEK-3):RENK DEĞİŞTİRİCİ

- **Buton** ve **h1** için HTML dosyamıza aşağıdaki eklemeleri yapabiliriz. Stillendirme için **style.css** dosyasını kullanabiliriz.

```
<h1 class="etiket">RENK:<span class="renk-yazı">"white"</span></h1>
<button class="btn renk">RENK DEĞİŞTİR</button>
```

- Renkleri bir dizide tanımlamak için;

```
const renkler = ["red","blue","green","yellow",black","pink","purple","brown"];
```

- Dizinin elaman sayısına göre rasgele bir sayı üretmek için;

```
const rasgeleSayı = Math.floor(Math.random() * renkler.length);
```

- Sayfa arka planına bu seçilen rasgele sayıyı aktarmak için;

```
document.querySelector("body").style.backgroundColor = renkler[rasgeleSayı];
```

DOM OLAYLARI (ÖRNEK-3):RENK DEĞİŞTİRİCİ

- Bu değişiklikler her buton tıklanışında olacağı için bir fonksiyon tanımlayabiliriz.

```
document.querySelector('.renk').addEventListener('click', () => {
  const renkler = ["red", "blue", "green", "yellow",
                   "black", "pink", "purple",'brown'];
  const rasgeleSayı = Math.floor(Math.random() * renkler.length);
  document.querySelector("body").style.backgroundColor = renkler[rasgeleSayı];
  document.querySelector(".renk-yazı").innerHTML = renkler[rasgeleSayı];
})
```

DOM OLAYLARI (ÖRNEK-4)

- Resimlerde görüldüğü gibi **AÇ** butonuna basıldığında **lamba_on** adındaki resmi, **KAPA** butonuna basıldığında ise **lamba_off** resmin gösteren uygulamayı yazalım

NOT: Bu örnekte stiller için **BOOTSTRAP** kütüphanesi kullanılmıştır.

Bootstrap kullanmak için kaynak klasörlerini projeye eklemek ve aşağıdaki gibi bağlantı kurmak gerekmektedir.

```
<link rel="stylesheet"
href="./css/bootstrap.css" />
```

DOM OLAYLARI (ÖRNEK-4)

index.html

```
<section>
 <img src="./img/lamba_off.gif" class="img" alt="lamba_off" />
 <div>
  <button type="button" class="btn-on btn-warning btn-lg">AÇ</button>
  <button type="button" class="btn-off btn-primary btn-lg">KAPA</button>
 </div>
 </section>
 <script src="app.js"></script>
</body>
```

app.js

```
document.querySelector(".btn-on").addEventListener("click", () => {
      document.querySelector(".img").src = "./img/lamba_on.gif";
});
document.querySelector(".btn-off").addEventListener("click", () => {
      document.querySelector(".img").src = "./img/lamba_off.gif";
});
```

DOM OLAYLARI (ÖRNEK-5)

- Bir önceki örneğe aşağıdaki gibi bir **<input> (text)** ve bir de **<input> (checkbox)** ekleyelim.

- Eğer **checbox tıklı** ise kutuya yazılan her karakteri **büyük harfe** çevirerek yazdırsın.

- Aksi durumda ise **küçük harfe** çevirerek yazdırsın.

DOM OLAYLARI (ÖRNEK-5)

```
<label for="checkbox">Küçük/Büyük</label>
<input class="checkbox" type="checkbox" />
```

index.html eklenecek kısım

```
document.querySelector(".textbox").addEventListener("keyup", () => {
        const text = document.querySelector(".textbox");
        const checkbox = document.querySelector(".checkbox");
        if (checkbox.checked) {
                text.value = text.value.toUpperCase();
        } else {
                text.value = text.value.toLowerCase();
        }
});
```

app.js

keyup: Klavyeden her bir tuş vuruşunda aktif olur.
value: input elemanının değerine erişmek için kullanılan özelliktir.
checked: Checkbox tıklı olduğunda **true** aksi durumda **false** değer döndürür.

YENİ ELEMAN OLUŞTURMA (ÖRNEK-6)

- Bir önceki örneğin en alt kısmına bir **<h1>** eklemesi yapalım. Ancak bunu **SADECE** Javascript kodu ile gerçekleştirelim.
  - h1'in yazısı '**Programlama Dilleri'**
  - Rengi kırmızı,
  - Üst margin'i %5 olsun

YENİ ELEMAN OLUŞTURMA (ÖRNEK-6)

1. İlk önce createElement() metodu ile bir **H1** elemanı oluşturmalıyız.
2. Daha sonra bu **H1** elemanı için createTextNode() metodu ile bir text düğümü oluşturmalıyız.
3. Bu text düğümünü appendChild() metodu ile **H1** elemanına bağlamalıyız.
4. H1 elemanını yine appendChild() metodu ile **body** elemanına (ya da yazının konmak istendiği başka elemana) bağlamalıyız.
5. En son olarak stil değişikliklerini yapabiliriz.

```
const h1 = document.createElement("h1");
const text = document.createTextNode("Programlama Dilleri");
h1.appendChild(text);
document.body.appendChild(h1);
h1.style.marginTop = "5%";
h1.style.color = "red";
```

YENİ ELEMAN OLUŞTURMA (ÖRNEK-6)

- Eğer **H1** elemanımızı body yerine bir başka elemana bağlamak istersek, body elemanına aşağıdaki gibi bir **div** elemanı ekleyebiliriz.

```
<div class='yeni-h1'></div>
```

- JS kodumuzu da aşağıdaki gibi değiştirebiliriz. Böylelikle, yeni **H1** elamanımızı HTML sayfasının istenilen noktasına yerleştirmemiz mümkün olacaktır.

```
const yeniH1 = document.querySelector(".yeni-h1");
const h1 = document.createElement("H1");
const text = document.createTextNode("Programlama Dilleri");
h1.appendChild(text);
yeniH1.appendChild(h1);
yeniH1.style.marginTop = "5%";
yeniH1.style.color = "red";
```

YENİ ELEMAN OLUŞTURMA (ÖRNEK-7)

- Bir önceki örnekteki **h1**'in altında aşağıdaki gibi **<li>**, **<input>**, **Ekle** ve bir de **Sil** adında bir **<button>** ekleyelim.
  - **input** alanına girilen bir yazı **Ekle** butonuna tıklandığında listenin sonuna eklenmeli ve **Sil** butonu ile de listenin en son elmanı silinmelidir.

YENİ ELEMAN OLUŞTURMA (ÖRNEK-7)

- Önce HTML dosyamıza **Listeyi**, **input'u** ve **Butonları** ekleyelim:

```
<div class='yeni-h1'></div>
<ul class="liste">
      <li>Javascript</li>
      <li>Java</li>
      <li>Phyton</li>
      <li>C++</li>
      <li>SQL</li>
</ul>
<input class="input-liste" type="text">
<button type="button" class="btn-ekle btn-success btn-lg">Ekle</button>
<button type="button" class="btn-sil btn-danger btn-lg">Sil</button>
```

YENİ ELEMAN OLUŞTURMA (ÖRNEK-7)

1. Ekle butonuna **her tıklandığında yeni bir liste elemanı** oluşturmamız,
2. Bu elemana ait bir **textNode** oluşturmamız ve bu textNode'un değerinin **input'dan** gelmesini sağlamamız,
3. Bu textNode'u yeni listeye **bağlamamız,**
4. Bu **yeni listeyi eski listeye** bağlamamız gerekmektedir.

```
document.querySelector(".btn-ekle").addEventListener("click", () => {
 const liste = document.querySelector(".liste");
 const inputListe = document.querySelector(".input-liste");

 const yeniListe = document.createElement("li");
 const textNode = document.createTextNode(inputListe.value);
 yeniListe.appendChild(textNode);
 liste.appendChild(yeniListe);
});
```

YENİ ELEMAN OLUŞTURMA (ÖRNEK-8)

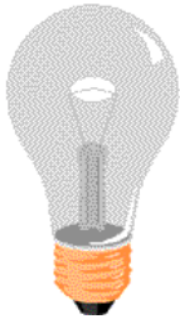
- Sil butonuna **her tıklandığında** **removeChild()** metodu ile listenin son elemanını silmeliyiz.

- Son eleman için **lastElementChild** değişkenin kullanabiliriz.

```
document.querySelector(".btn-sil").addEventListener("click", () => {
 const liste = document.querySelector(".liste");
 liste.removeChild(liste.lastElementChild)
});
```

YENİ ELEMAN OLUŞTURMA (ÖRNEK-8)

Örnekler



AÇ KAPA

Bir şeyler yazınız Küçük/Büyük

Programlama Dilleri

- **Javascript**
- **Java**
- **Phyton**
- **C++**
- **SQL**

Ekle Sil