



Национална програма
"Обучение за ИТ умения и кариера"
<https://it-kariera.mon.bg/e-learning/>

Министерството на
образованието и науката
<https://www.mon.bg>

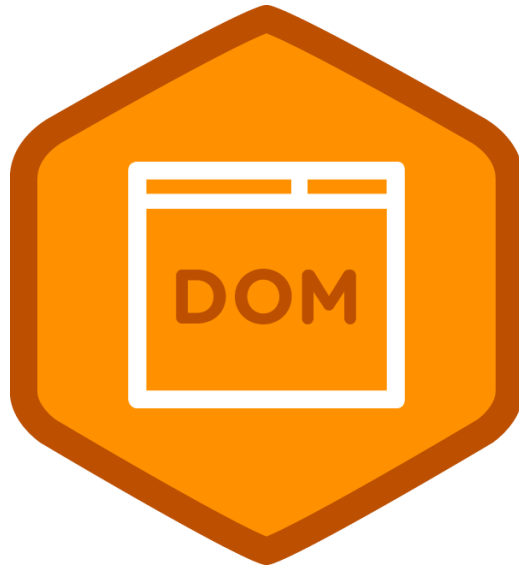


DOM

Document Object Model

Съдържание

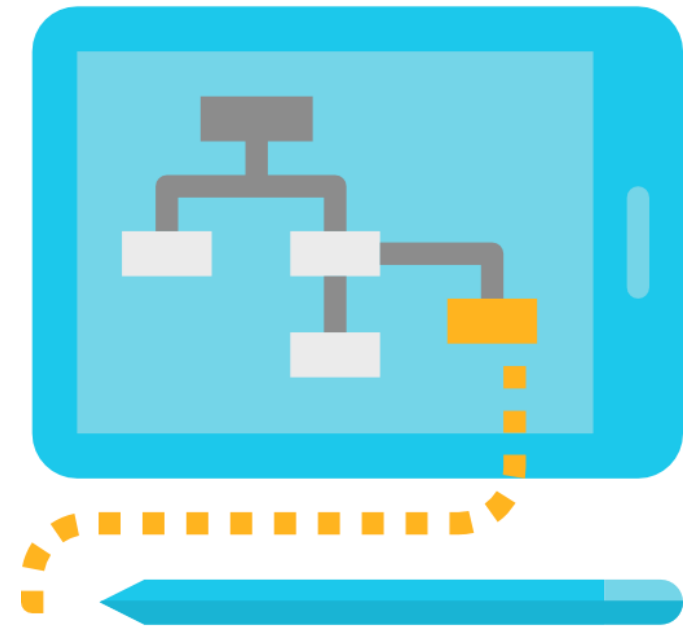
1. Какво е DOM?
2. Манипулиране на DOM
3. Родители и Деца Елементи
4. DOM Събития



Document Object Model

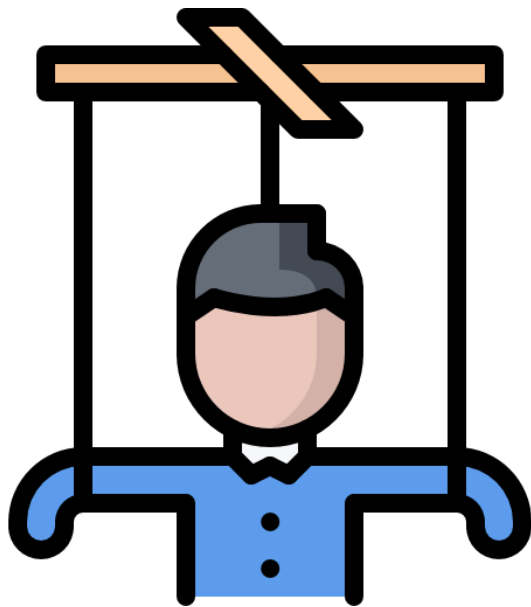
Document Object Model

- **DOM** представя документ като възли и обекти
 - По този начин езиките за програмиране могат да се свързват със страниците
- **DOM** е стандарт за:
 - **Вземане** на HTML елемент
 - **Промяна** на HTML елемент
 - **Добавяне** на HTML елемент
 - **Изтриване** на HTML елемент



HTML DOM

- **HTML DOM** е модел от обекти за **HTML**.
- Той определя:
 - HTML елементи като **обекти**
 - **Свойства** за всички HTML елементи
 - **Методи** за всички HTML елементи
 - **Събития** за всички HTML елементи



Манипулиране на DOM

Промяна на DOM дървото

Избор на елементи

- Има няколко начина за намиране на определен HTML елемент в DOM:
 - По id - `getElementById()`
 - По име на таг - `getElementsByTagName()`
 - По име на клас - `getElementsByClassName()`
 - От CSS селектор - `querySelector()`

CSS Селектори

- CSS селекторите са низове, които следват CSS синтаксиса за съвпадение
- Те позволяват много бързо и мощно съчетаване на елементи, например:
 - `#main` - връща елемента с ID "main"
 - `#content div` - избира всички `<div>` елементи в #content
 - `.note, .alert` - всички елементи с клас "note" или "alert"
 - `input[name = 'login']` - `<input>` с име "login"

DOM манипулации [1/2]

- HTML DOM позволява на JavaScript да променя съдържанието на HTML елементи:
 - `innerHTML`
 - `attributes`
 - `setAttribute()`
 - `style.property`

DOM манипулации [2/2]

- Можем да създаваме, добавяме и премахваме HTML елементи динамично:
 - `removeChild()`
 - `appendChild()`
 - `replaceChild()`
 - `document.write()`

Създаване на DOM елементи

- Създаване на нов DOM елемент

```
let p = document.createElement("p");  
let li = document.createElement("li");
```

- Създаване на копие на DOM елемент

```
let li = document.getElementById("my-list");  
let newLi = li.cloneNode(true);
```

- Горният код създава нови елементи
- Тези елементи не съществуват никъде, освен като стойности в променливите

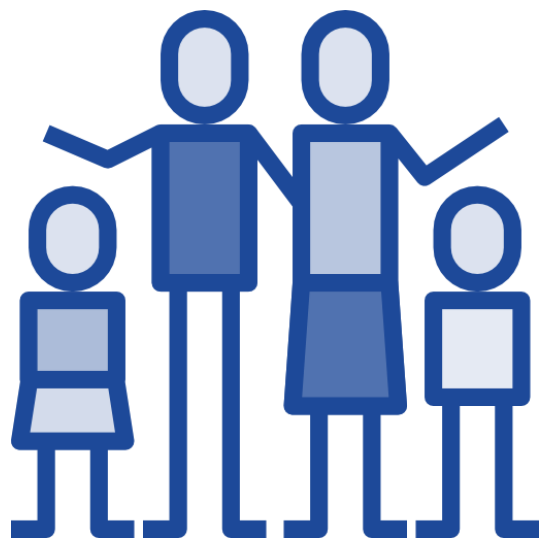
Изтриване на DOM елементи

```
<div id="div1">  
  <p id="p1">This is a paragraph.</p>  
  <p id="p2">This is another paragraph.</p>  
</div>
```

```
let parent = document.getElementById("div1");  
let firstChild = document.getElementById("p1");  
let secondChild = document.getElementById("p2");  
  
firstChild.remove();  
parent.removeChild(secondChild);
```

Добавяне на DOM елементи

```
let list = document.createElement("ul");  
let firstLi = document.createElement("li");  
firstLi.textContent = "Peter";  
list.appendChild(firstLi);  
let secondLi = document.createElement("li");  
secondLi.innerHTML = "<b>Maria</b>";  
list.appendChild(secondLi);  
document.body.appendChild(list);
```



Родители и Деца Елементи

Родители и Деца Елементи

- Всеки DOM елемент има родител
- Родителите могат да се достъпят с ключовите думи **.parent** или **.parentNode**

```
▼ <div>  
  <p>This is a paragraph.</p>  
  <p>This is another paragraph.</p>  
</div>
```

```
let firstP = document.getElementsByTagName('p')[0];  
console.log(firstP.parent);
```

```
▶ <div>...</div>
```

Родители и Деца Елементи

- Когато някой елемент съдържа други елементи, това означава, че той е родител на тези елементи
- Също така тези елементи са деца на родителя
 - Те могат да бъдат достъпени по ключова дума **.children**

```
▼ <div>  
  <p>This is a paragraph.</p>  
  <p>This is another paragraph.</p>  
</div>
```

```
▼ HTMLCollection(2) [p, p]  
  ► 0: p  
  ► 1: p  
  length: 2
```

```
let pElements = document.getElementsByTagName('div')  
[0].children;
```




DOM Събития

Работа с DOM събития

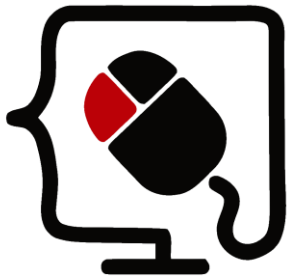
DOM Събития

- Събитията са действия или явления
- Те позволяват на JavaScript да задава обработващи събития на различни елементи
- Събитията обикновено се използват в комбинация с функции
 - Функцията няма да се изпълни преди да се случи събитието

```
htmlRef.addEventListener('click', handler);
```

Обобщение

- **DOM** е програмен API за HTML и XML документи
- **DOM** манипулации
- **Родители** и **Деца** Елементи
 - Всеки **DOM** елемент има **родител**
- DOM **Събития**



Национална програма
"Обучение за ИТ умения и кариера"
<https://it-kariera.mon.bg>

Министерството на
образованието и науката
<https://www.mon.bg>



Документът е разработен за нуждите на Национална програма "Обучение за ИТ умения и кариера" на Министерството на образованието и науката (МОН) и се разпространява под свободен лиценз CC-BY-NC-SA (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).