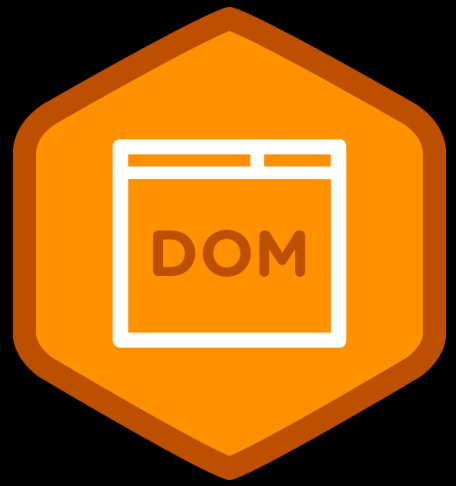
Работа с DOM

Document Object Model



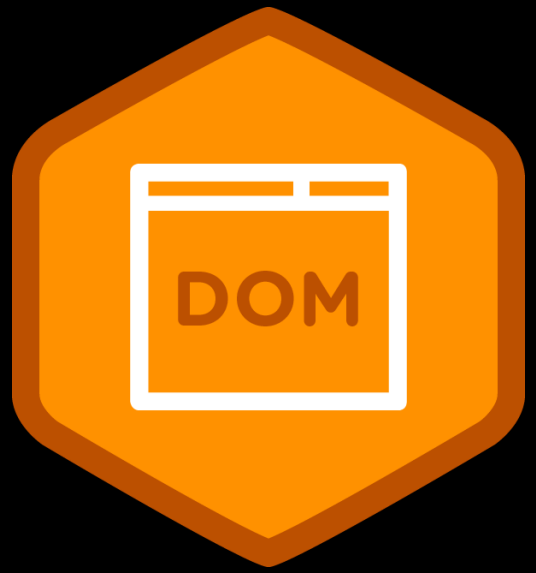
Учителски екип

Обучение за ИТ кариера

https://it-kariera.mon.bg/e-learning/

Съдържание

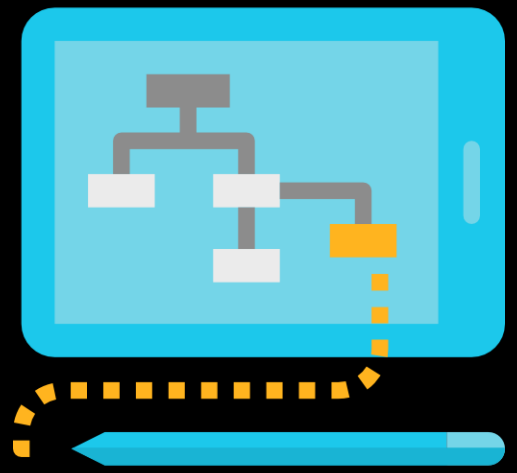
1. Какво е DOM?
2. Манипулиране на DOM
3. Родители и Деца Елементи
4. DOM Събития



Document Object Model

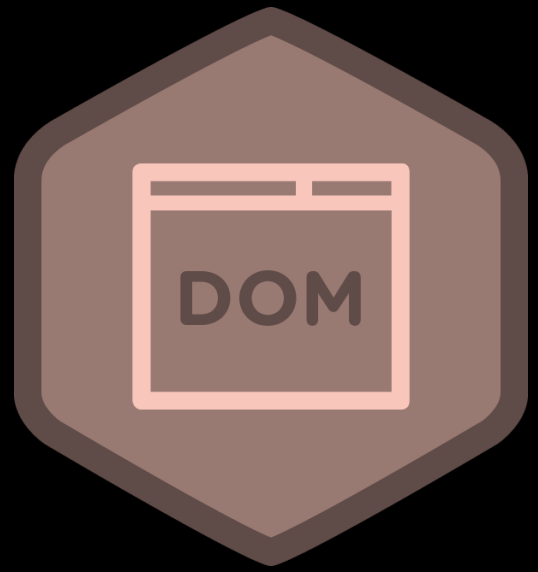
Document Object Model

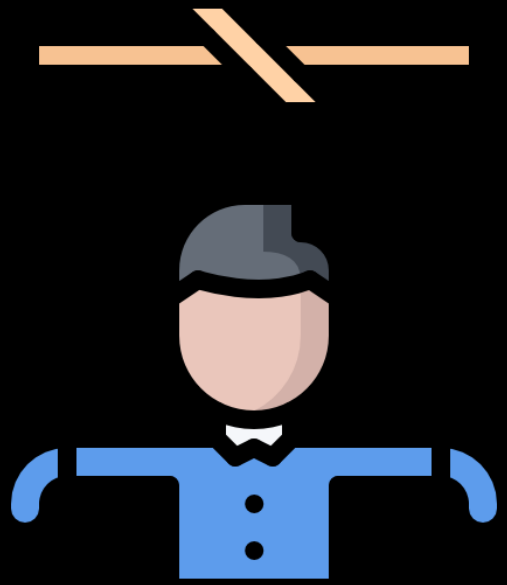
- **DOM** представя документ като възли и обекти
  - По този начин езиците за програмиране могат да се свързват със страниците
- **DOM** е стандарт за:
  - **Вземане** на HTML елемент
  - **Промяна** на HTML елемент
  - **Добавете** на HTML елемент
  - **Изтриване** на HTML елемент



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 ** ** елементи в #content
  - **".note, .alert"** - всички елементи с клас "note" или "alert"
  - **"input [name = 'login']"** - **<input>** с име "login"

DOM манипулации

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

DOM манипулации

- 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);
```

Създаване 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 имат достъп 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>
```

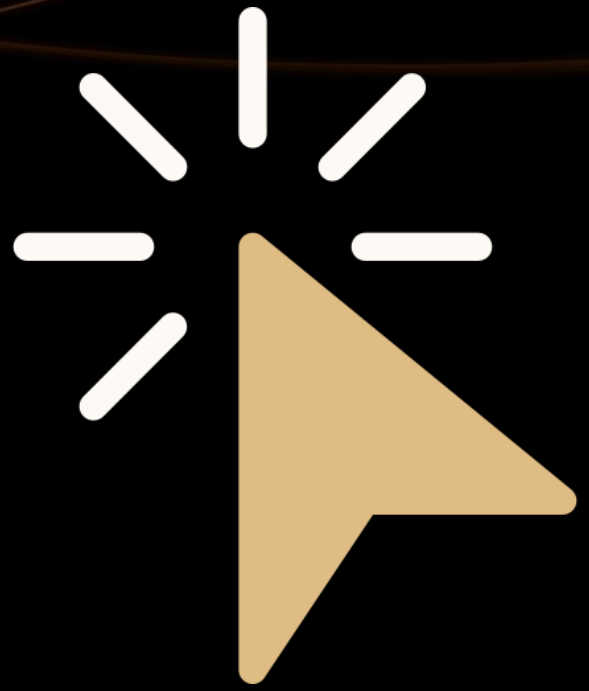
Родители и Деца Елементи

- 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 достъпни 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

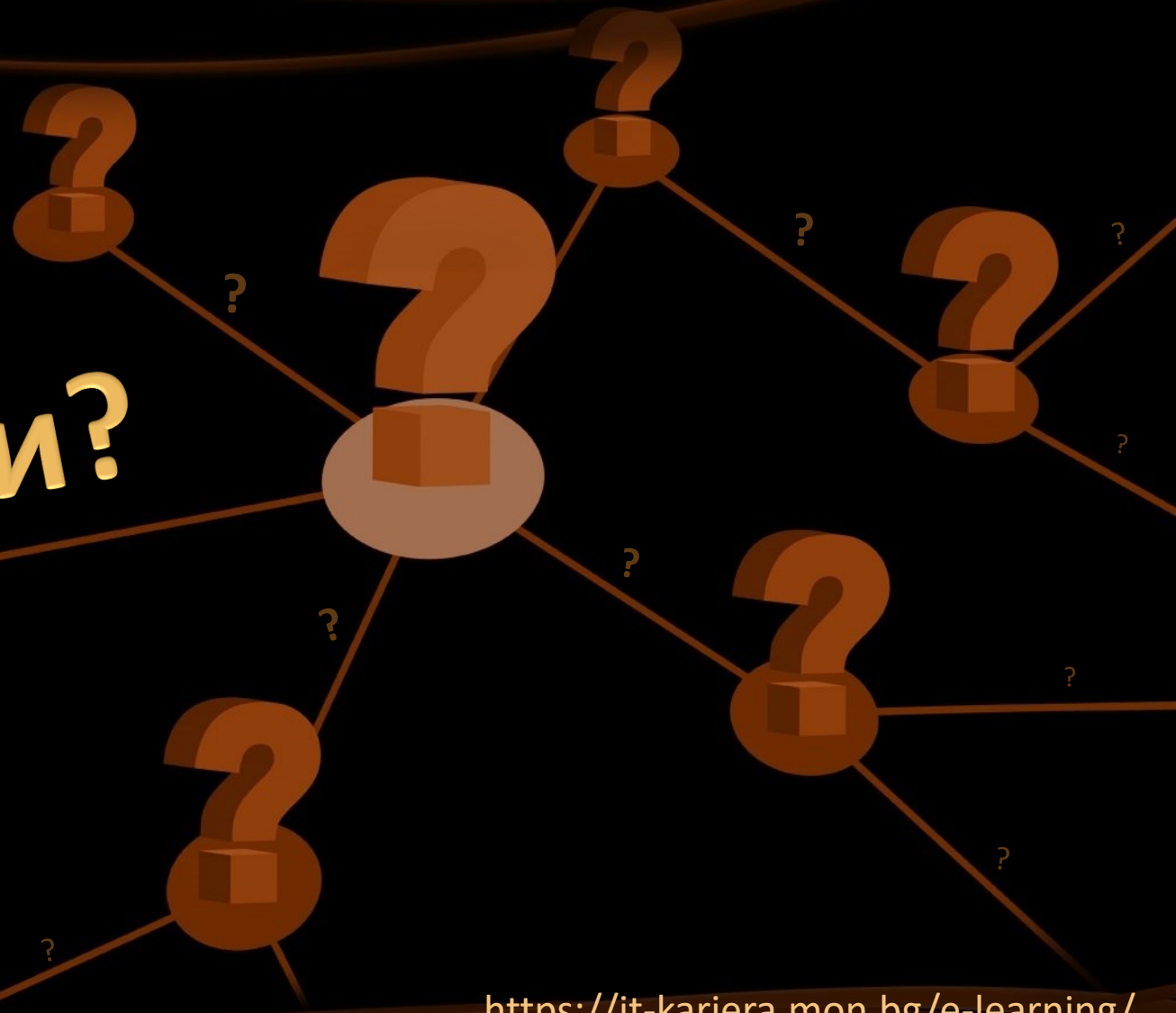
```
htmlRef.addEventListener('click', handler);
```

Какво научихме в този час?

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

Работа с DOM

Въпроси?



Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**