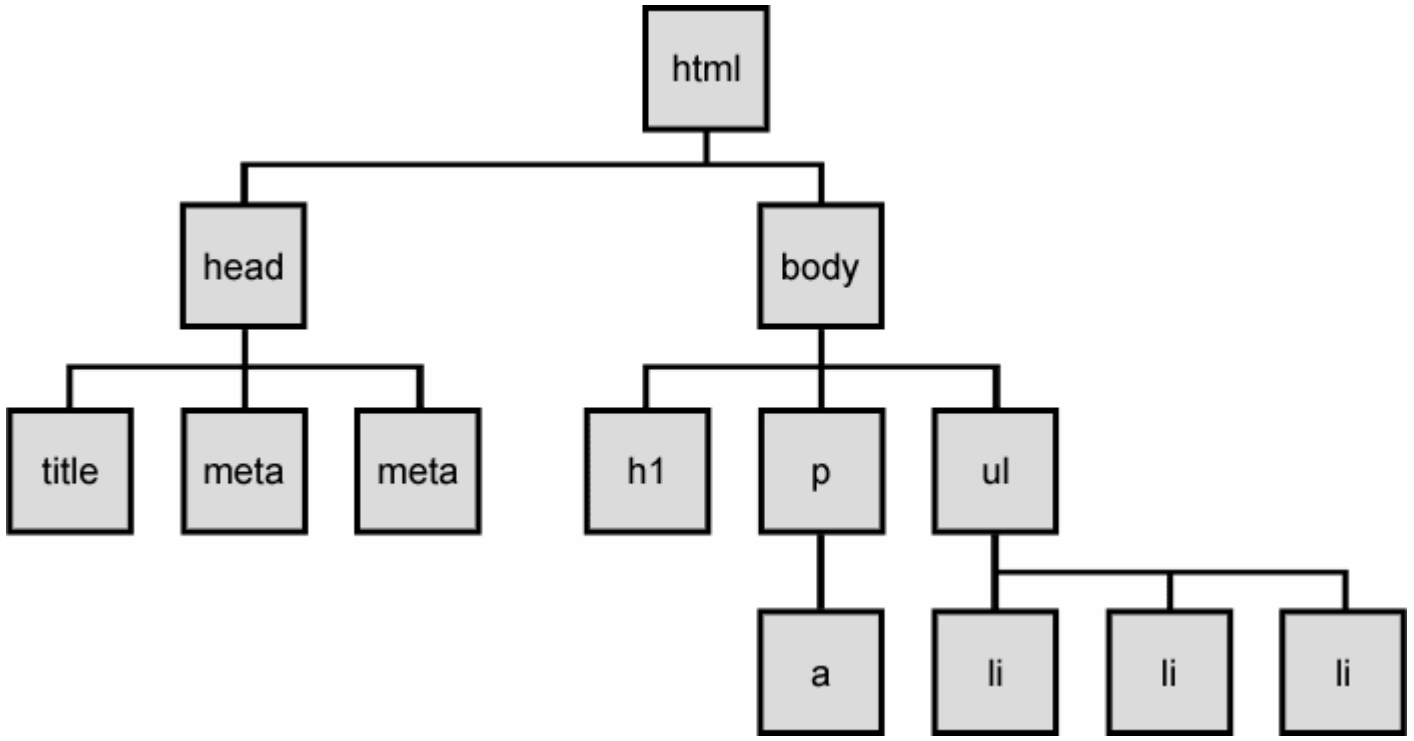
Programowanie w JavaScript

WSEI

#szkoła programowania

DOM

DOM – document object model – w JS reprezentowany poprzez obiekt document skąd zaczynamy wszystkie akcje wyszukiwania elementów.



DOM – wyszukiwanie pojedyncze

W JS mamy kilka metod wyszukiwania elementów na naszej stronie. Możemy je podzielić na dwie kategorie:

- Takie które zwracają pojedynczy element
- Takie które zwracają kolekcję elementów czyli tablicę

Wyszukiwanie pojedynczego elementu:

```
document.querySelector('#selector');
document.getElementById('selector');
```

Wyszukiwanie kolekcji elementów:

```
document.querySelectorAll('.selector');

document.getElementsByTagName('div');

document.getElementsByClassName('selector');
```

Kilka przykładów:

```
const buttonRed = document.querySelector('div.button button.btn');

const spansPink = document.querySelectorAll('span.pink');

const link = document.getElementById('link');

const table = document.getElementById('my-table');

const rowInTable = table.querySelectorAll('tr');
```

DOM – elementy

Elementy na stronie wyszukujemy aby potem coś z nimi zrobić. Na przykład usunąć, ukryć, zmienić tekst, pobrać tekst, pobrać dane z atrybutów itp.

Każdy znacznik (tag) html posiada własne atrybuty które mają odzwierciedlenie w JS. Poniżej lista najczęściej używanych:

classList – zwraca pseudotablicę class danego elementu

className – zwraca lub ustawia klasę na danym elemencie

id – zwraca lub ustawia id danego elementu

innerHTML – zwraca lub ustawia kod HTML danego elementu

outerHTML – zwraca lub ustawia kod HTML danego elementu wraz z tym elementem

innerText – zwraca lub ustawia tekst danego elementu

tagName – zwraca nazwę tagu

dataset – zwraca obiekt dataset z atrybutu data elementu

children – zwraca pseudotablicę z elementami wewnętrznymi (dziećmi) elementu

parentElement – zwraca element nadrzędny (rodzica)

```
<div class="some-div" id="divId" data-first="1" data-second="2">
    Główny tekst
    <span>Jakiś tekst</span>
</div>

const myDiv = document.querySelector('#divId');

myDiv.className; //some-div
myDiv.id; // divId
myDiv.innerHtml; // <span>Jakiś tekst</span>
myDiv.outerHTML; /* <div class="some-div" id="divId" data-first="1" data-second="2">
                        Główny tekst
                        <span>Jakiś tekst</span>
                    </div> */
myDiv.innerText; //Główny tekst
myDiv.tagName; // DIV
myDiv.dataset; // {first: "1", second: "2"}
```

DOM – classList

classList zwraca listę klas elementu. Jednak jest to o wiele bardziej złożony obiekt.

classList.add(className) – dodaje klase do elementu

classList.remove(className) – usuwa klasę z elementu

classList.toggle(className) – przełącza klasę na elemencie

```
<div id="button" class="class other hmm"></div>

const myButton = document.querySelector('#button');

myButton.classList // ['class', 'other', 'hmmm']

myButton.className // 'class other hmm

myButton.classList.add('newClassName')

myButton.classList.remove('other');

myButton.classList.toggle('toggleClassName');
```

Dataset oprócz zwracania obiektu z atrybutów data danego elementu może również ustawiać takie atrybuty oraz zmieniać

```
<div id="datasetChecker" data-user="Mateusz"></div>

const datasetTest = document.querySelector('#datasetChecker');

datasetTest.dataset; // {user: "Mateusz"}

datasetTest.dataset.user; // Mateusz

datasetTest.dataset.user = 'Alek'; //zmienia Mateusz na Alek

datasetTest.dataset.newValue = 'new value'; // dodaje nowy data atrybut z wartością new value
```

Przy pomocy JS możemy dodawać, edytować, usuwać wszystkie atrybuty danego elementu.

```
<a href="https://google.pl" id="googleLink">Google</a>

const link = document.querySelector('#googleLink');

link.hasAttribute('href'); // true

link.getAttribute('href'); // https://google.pl

link.removeAttribute('href'); // usuwa atrybut href

link.setAttribute('href', 'https://google.com'); //dodaje lub zmienia atrybut
```

Obiekt style pozwala na dodawanie i modyfikowanie stylowania z poziomu JS. Jest to jednak bardzo nieoptymalne i nie zaleca się robienia tego bezpośrednio z JS. Lepszą metodą jest dodanie odpowiednich styli w plikach css a następnie przy pomocy JS zmiana odpowiednich klas na elemencie które będą zmieniać wygląd elementu.

Obiekt style może modyfikować tylko style które zostały dodane przy pomocy JS. Nie ma wpływu na plik css

```
<a href="https://google.pl" id="googleLink">Google</a>

const link = document.querySelector('#googleLink');

link.style.backgroundColor; // zwraca kolor backgroundu

link.style.backgroundColor = 'red'; // ustawia background na red
```