Scenariusz Lekcji: Hierarchia obiektów DOM w JavaScript

Wiedza wstępna: Uczniowie mają podstawową wiedzę z HTML, CSS oraz podstawy programowania w JavaScript.

Środki dydaktyczne: Komputer z dostępem do internetu, edytor kodu (np. Visual Studio Code), projektor, tablica interaktywna, plansze z diagramami DOM, kolorowe karteczki.

Cele lekcji

1. Cel ogólny: Uczniowie zrozumieją hierarchię obiektów DOM (Document Object Model) i nauczą się manipulować elementami DOM za pomocą JavaScript.

2. Cele szczegółowe:

Wiedza: Uczniowie poznają strukturę DOM i zrozumieją, jak HTML jest reprezentowany w postaci drzewa obiektów.

Umiejętności: Uczniowie będą potrafili uzyskiwać dostęp do elementów DOM, zmieniać ich właściwości oraz tworzyć i usuwać elementy w DOM.

Postawy: Uczniowie docenią znaczenie manipulacji DOM w tworzeniu dynamicznych i interaktywnych stron internetowych.

Przebieg lekcji

1. Wprowadzenie

- Powitanie i sprawdzenie obecności.
- Przedstawienie tematu i celów lekcji:

"Dzisiaj omówimy hierarchię obiektów DOM w JavaScript. DOM, czyli Document Object Model, to interfejs, który pozwala programom na dostęp i modyfikację zawartości, struktury oraz stylów dokumentów HTML. Zrozumienie DOM jest kluczowe do tworzenia dynamicznych i interaktywnych stron internetowych."

Krótka ankieta sprawdzająca wstępną wiedzę uczniów:

"Czy ktoś z Was wie, czym jest DOM? Może ktoś próbował już manipulować elementami HTML za pomocą JavaScript?"

2. Wykład interaktywny

- Omówienie struktury DOM:

"DOM reprezentuje dokument HTML jako hierarchiczne drzewo obiektów. Każdy element HTML to węzeł drzewa. Najwyższym węzłem jest document, który zawiera węzeł html, a ten z kolei zawiera head i body."

Diagram hierarchii DOM: Pokaz slajdu z diagramem przedstawiającym strukturę drzewa DOM.

"Każdy węzeł może mieć swoje dzieci (child nodes) i rodzica (parent node). Na przykład, body jest dzieckiem html, a rodzicem dla wielu elementów, takich jak div, p, a itp."

- Przykłady dostępu do elementów DOM:

"Możemy uzyskiwać dostęp do elementów DOM za pomocą różnych metod, takich jak getElementById, getElementsByClassName, getElementsByTagName, querySelector i querySelectorAll."

Przykład:

```
javascript

const element = document.getElementById('myElement');

const elements = document.getElementsByClassName('myClass');

const paragraphs = document.getElementsByTagName('p');

const firstElement = document.querySelector('.myClass');

const allElements = document.querySelectorAll('.myClass');
```

- Manipulacja elementami DOM:

"Możemy zmieniać właściwości elementów DOM, takie jak innerHTML, style, attributes i inne."

Przykład:

```
javascript

const element = document.getElementById('myElement');

element.innerHTML = 'Hello, World!';

element.style.color = 'blue';

element.setAttribute('datacustom', 'value');
```

- Tworzenie i usuwanie elementów DOM:

"Możemy tworzyć nowe elementy za pomocą document.createElement, a następnie dodawać je do drzewa DOM za pomocą appendChild lub insertBefore."

Przykład:

```
javascript

const newElement = document.createElement('div');

newElement.innerHTML = 'New Element';

document.body.appendChild(newElement);
```

"Elementy mogą być również usuwane za pomocą removeChild."

Przykład:

```
javascript

const parent = document.getElementById('parentElement');
```

```
const child = document.getElementById('childElement');
parent.removeChild(child);
```

3. Pokaz

- Prezentacja na żywo:

"Teraz pokażę wam, jak manipulować DOM w praktyce. Utworzymy prostą stronę HTML z listą elementów i przyciskiem do dodawania nowych elementów do listy."

Kod HTML:

```
html
<!DOCTYPE html>
<html lang="pl">
<head>
    <meta charset="UTF8">
    <title>Manipulacja DOM</title>
</head>
<body>
    <ul id="myList">
        <li>Element 1</li>
        <li>Element 2</li>
    </ul>
    <button onclick="addItem()">Dodaj element</button>
    <script src="script.js"></script>
</body>
</html>
```

Kod JavaScript (script.js):

```
javascript

function addItem() {
    const newItem = document.createElement('li');
    newItem.innerHTML = 'Nowy Element';
    document.getElementById('myList').appendChild(newItem);
}
```

4. Ćwiczenia praktyczne

Zadanie 1: Napisanie skryptu, który zmienia tekst i styl wybranego elementu.

"Napiszcie skrypt, który zmienia tekst paragrafu o id myParagraph na 'Zmieniony tekst' i ustawia kolor tekstu na czerwony."

Zadanie 2: Tworzenie skryptu, który dodaje nowy element do listy po kliknięciu przycisku.

"Stwórzcie przycisk, który po kliknięciu dodaje nowy element do listy ul."

Zadanie 3: Praca w grupach stworzenie interaktywnej galerii obrazów, która pozwala na dodawanie, usuwanie i edytowanie opisów obrazów.

"W grupach stwórzcie interaktywną galerię, która pozwala użytkownikowi na dodawanie nowych obrazów z opisami, usuwanie wybranych obrazów oraz edytowanie opisów."

5. Dyskusja i analiza

Omówienie rozwiązań zadań:

"Kto chciałby zaprezentować swoje rozwiązanie pierwszego zadania? Pokażcie kod i wyjaśnijcie, jak działa."

Dyskusja nad napotkanymi problemami i sposobami ich rozwiązania:

"Jakie problemy napotkaliście podczas wykonywania zadań? Jak je rozwiązaliście? Czy macie jakieś pytania dotyczące manipulacji DOM?"

6. Podsumowanie

Podsumowanie kluczowych punktów lekcji:

"Dzisiaj nauczyliśmy się, czym jest DOM i jak możemy manipulować jego elementami za pomocą JavaScript. Hierarchia DOM pozwala nam na łatwy dostęp do różnych części dokumentu HTML i dynamiczne zmienianie jego struktury i wyglądu."

Pytania i odpowiedzi:

"Czy macie jeszcze jakieś pytania na temat hierarchii obiektów DOM i manipulacji DOM w JavaScript?"