Об'єктна модель документа

План:

1. Загальна характеристика Document Object Model (DOM) як інструмента роботи і динамічних змін на сторінці.
2. Властивості та методи доступу до елементів DOM.
3. Призначення DOM-колекцій та їх роль у взаємодії з елементами веб-сторінки.
4. Типи DOM-колекцій та їх особливості.

Загальна характеристика Document Object Model (DOM) як інструмента роботи і динамічних змін на сторінці.

Document Object Model (DOM) - це стандартна об'єктна модель, яка представляє структуру та вміст документа веб-сторінки. DOM надає програмам доступ до елементів сторінки та дозволяє змінювати їх структуру, стиль та вміст з використанням мов програмування, таких як JavaScript.

DOM відображає HTML-документ у деревоподібну структуру, де кожен елемент сторінки стає вузлом у цьому дереві.

Роль DOM у веб-розробці полягає в тому, що він надає програмам доступ до структури та елементів веб-сторінки, що дозволяє змінювати їх динамічно, додавати нові елементи, видаляти чи змінювати наявні.

DOM дозволяє веб-розробникам створювати інтерактивні сторінки, реагуючи на події користувача, змінюючи вміст сторінки, анімуючи елементи та багато іншого.

Один з найпоширеніших способів використання DOM у веб-розробці - це за допомогою мови програмування JavaScript. JavaScript надає доступ до DOM через свою вбудовану об'єктну модель.

За допомогою JavaScript можна звертатися до елементів сторінки, зчитувати та змінювати їх властивості, додавати події та реагувати на них, створювати нові елементи, видаляти чи переміщувати існуючі елементи і багато іншого. Це дає веб-розробникам гнучкість та можливості для створення багатофункціональних веб-додатків та взаємодії з користувачем.

DOM також використовується для роботи з XML-документами. Він дає можливість програмам зчитувати та змінювати XML-структуру, виконувати пошук елементів, створювати нові елементи та зв'язувати їх з існуючою структурою документа.

Основна ідея застосування DOM у веб-розробці полягає в тому, що весь контент сторінки стає доступним та маніпульованим через об'єктну модель, що забезпечує зручну та послідовну роботу з елементами сторінки.

Веб-розробники можуть маніпулювати DOM-елементами, створювати, зчитувати та змінювати їх, створювати динамічний вміст та

забезпечувати взаємодію з користувачем. Все це робить DOM важливою складовою інструментарію веб-розробки та дозволяє створювати потужні та інтерактивні веб-додатки.

Один з основних принципів DOM - це доступність до елементів та їхніх властивостей через програмний інтерфейс. Це означає, що розробники можуть отримати доступ до будь-якого елемента сторінки, зчитати та змінювати його властивості, додавати, видаляти або змінювати елементи, а також здійснювати інші операції, що впливають на вигляд і поведінку сторінки.

DOM є потужним інструментом для роботи зі сторінками веб-сайтів. Він дозволяє створювати динамічні зміни на сторінці, змінювати вміст та вигляд елементів, реагувати на події користувача та забезпечувати взаємодію з додатками.

Наприклад, веб-розробники можуть застосовувати DOM для виконання наступних завдань:

Зміна вмісту сторінки: DOM дозволяє динамічно змінювати текст, зображення та інші елементи сторінки. Розробники можуть змінювати вміст елементів, додавати нові елементи, видаляти існуючі або змінювати їхні атрибути.

Маніпулювання стилями і виглядом: з використанням DOM можна змінювати CSS-стилі елементів сторінки, надавати їм нові класи, змінювати розміри, кольори, положення та інші атрибути, що впливають на їх зовнішній вигляд.

Взаємодія з користувачем: DOM дозволяє реагувати на події, спричинені користувачем, такі як клік мишею, натискання клавіші, наведення курсора тощо. Розробники можуть створювати обробники подій, що реагують на ці дії користувача та виконують певні дії, наприклад, відправляють дані на сервер або анімують елементи сторінки.

Динамічне створення та модифікація елементів: DOM дозволяє створювати нові елементи та додавати їх до сторінки, як під елементи існуючих елементів або як самостійні елементи. Це дозволяє створювати динамічні структури сторінки та додавати новий вміст на льоту.

DOM є важливим інструментом у веб-розробці, оскільки він дає можливість розробникам створювати інтерактивні та динамічні веб-додатки. Він надає доступ до структури сторінки, її елементів та властивостей, що дозволяє змінювати їх за допомогою програмного коду.

DOM використовується у багатьох технологіях веб-розробки, таких як JavaScript, jQuery, AJAX, і становить основу для багатьох інших інструментів і фреймворків.

Властивості та методи доступу до елементів DOM.

Об'єктна модель документа (DOM) - це стандартна програмний інтерфейс, який надає доступ до структури та вмісту веб-документа.

DOM представляє документ у вигляді дерева об'єктів, де кожен елемент, текстовий вузол, атрибут та інші елементи сторінки представлені як об'єкти з властивостями і методами.

Властивості та методи доступу до елементів DOM дозволяють розробникам маніпулювати вмістом, стилями, атрибутами та подіями елементів веб-сторінки. Декілька основних властивостей та методів доступу до елементів DOM включають наступне:

1. getElementById(): Цей метод дозволяє отримати посилання на елемент сторінки за його унікальним ідентифікатором (ID). Використовується таким чином:

var element = document.getElementById("elementId");

2. getElementsByClassName(): Цей метод повертає колекцію елементів, які мають заданий клас. Використовується так:

var elements = document.getElementsByClassName("className");

3. getElementsByTagName(): Цей метод повертає колекцію елементів зазначеного типу тегу. Наприклад, можна отримати всі елементи <p> на сторінці:

var paragraphs = document.getElementsByTagName("p");

4. querySelector(): Цей метод повертає перший елемент, що відповідає заданому CSS-селектору. Використовується для отримання елементів за класом, тегом, ID або іншими властивостями:

var element = document.querySelector("#elementId");

5. querySelectorAll(): Цей метод повертає всі елементи, які відповідають заданому CSS-селектору. Використовується так:

var elements = document.querySelectorAll(".className");

6. textContent: Ця властивість дозволяє отримати або встановити текстовий вміст елемента. Наприклад, можна отримати текст з <p> елемента:

var text = element.textContent;

7. innerHTML: Ця властивість дозволяє отримати або встановити HTML-вміст елемента. Можна вставляти HTML-код в елемент або отримувати його з нього:

var htmlContent = element.innerHTML;

8. setAttribute(): Цей метод дозволяє встановити значення атрибута елемента. Наприклад, можна встановити атрибут src для зображення:

element.setAttribute("src", "image.jpg");

9. addEventListener(): Цей метод додає обробник подій до елемента. Використовується для визначення реакції на події, такі як клік, наведення курсора, введення тексту та інші:

```
element.addEventListener("click", function() {
  // дії, що виконуються при кліку
});
```

Призначення DOM-колекцій та їх роль у взаємодії з елементами веб-сторінки.

DOM-колекції є важливою частиною Об'єктної моделі документа (DOM) і використовуються для зберігання і доступу до груп елементів веб-сторінки. Вони відображаються як списки або колекції об'єктів і надають розробникам зручний спосіб взаємодії з багатьма елементами одночасно.

Кожна колекція має свої властивості та методи, які дозволяють виконувати різні операції над елементами.

Основне призначення DOM-колекцій полягає в тому, що вони дозволяють розробникам маніпулювати групами елементів одночасно. Це особливо корисно, коли потрібно здійснити масові зміни або отримати інформацію з декількох елементів одночасно.

Роль DOM-колекцій полягає в тому, що вони дозволяють розробникам ефективно маніпулювати групами елементів. Завдяки колекціям можна змінювати стилі, додавати або видаляти класи, збирати значення полів форми, здійснювати перебір елементів та багато іншого. Вони дозволяють скоротити код і зберегти час при взаємодії з великою кількістю елементів на веб-сторінці.

Наприклад, за допомогою цих колекцій можна змінити текст у всіх <p> елементах одночасно, пройшовшись по колекції і встановивши нове значення textContent для кожного елемента:

```
var paragraphs = document.getElementsByTagName("p");
for (var i = 0; i < paragraphs.length; i++) {
  paragraphs[i].textContent = "Новий текст";
```

}

Отже, DOM-колекції грають важливу роль у взаємодії з елементами веб-сторінки, надаючи зручний спосіб доступу до груп елементів та забезпечуючи можливості для їх маніпулювання та обробки.

Типи DOM-колекцій та їх особливості.

Об'єктна модель документа (DOM) надає розробникам доступ до веб-сторінки шляхом представлення її елементів у вигляді колекцій. DOM-колекції є наборами елементів, які можна маніпулювати та використовувати для зміни структури, стилів та властивостей сторінки.

У DOM існує кілька типів колекцій, кожен з яких має свої особливості та методи доступу до елементів. Давайте розглянемо декілька основних типів DOM-колекцій та їх характеристики:

- HTMLCollection: Цей тип колекції представляє групу елементів, які є дітьми певного батьківського елемента або є результатом запиту getElementsByTagName() або getElementsByClassName(). Основна особливість HTMLCollection полягає в тому, що вона автоматично оновлюється при зміні структури документа. Наприклад, якщо додати новий елемент до колекції, вона буде автоматично оновлена. HTMLCollection можна ітерувати за допомогою циклу for або отримати доступ до елемента за допомогою індекса.

- NodeList: Цей тип колекції представляє результат запиту querySelectorAll() і містить список елементів, які відповідають заданому CSS-селектору. NodeList також автоматично оновлюється при зміні структури документа. Одна з особливостей NodeList полягає в тому, що вона підтримує ітерацію за допомогою циклу for...of та методів, які є у сучасних масивах, таких як forEach(), map(), filter() та інші.
- HTMLCollectionOf: Цей тип колекції є підтипом HTMLCollection і представляє колекцію елементів з конкретним ім'ям тегу. Наприклад, document.images є HTMLCollectionOf зі зображеннями (<img> елементами). Він має ті ж властивості та методи, що і звичайний HTMLCollection.
- NamedNodeMap: Цей тип колекції представляє атрибути елемента, які можна отримати за допомогою імені. Наприклад, атрибути елемента доступні через властивість attributes. NamedNodeMap можна ітерувати за допомогою циклу for або отримати доступ до атрибута за допомогою імені.
- DOMTokenList: Цей тип колекції представляє список класів елемента, які можна маніпулювати за допомогою методів, таких як add(), remove(), toggle() та інші. Він дозволяє додавати, видаляти та перевіряти наявність класів в елементі.

Кожен тип колекції має свої унікальні методи та особливості, які дозволяють ефективно працювати з елементами веб-сторінки.

Розробники можуть використовувати ці колекції для знаходження, маніпулювання та зміни властивостей елементів з використанням зручного інтерфейсу, що надає DOM.