hillel IT school

Lesson 18

План заняття

- Властивості вузлів;
- Внесення змін у документ;
- Стилі та класи;

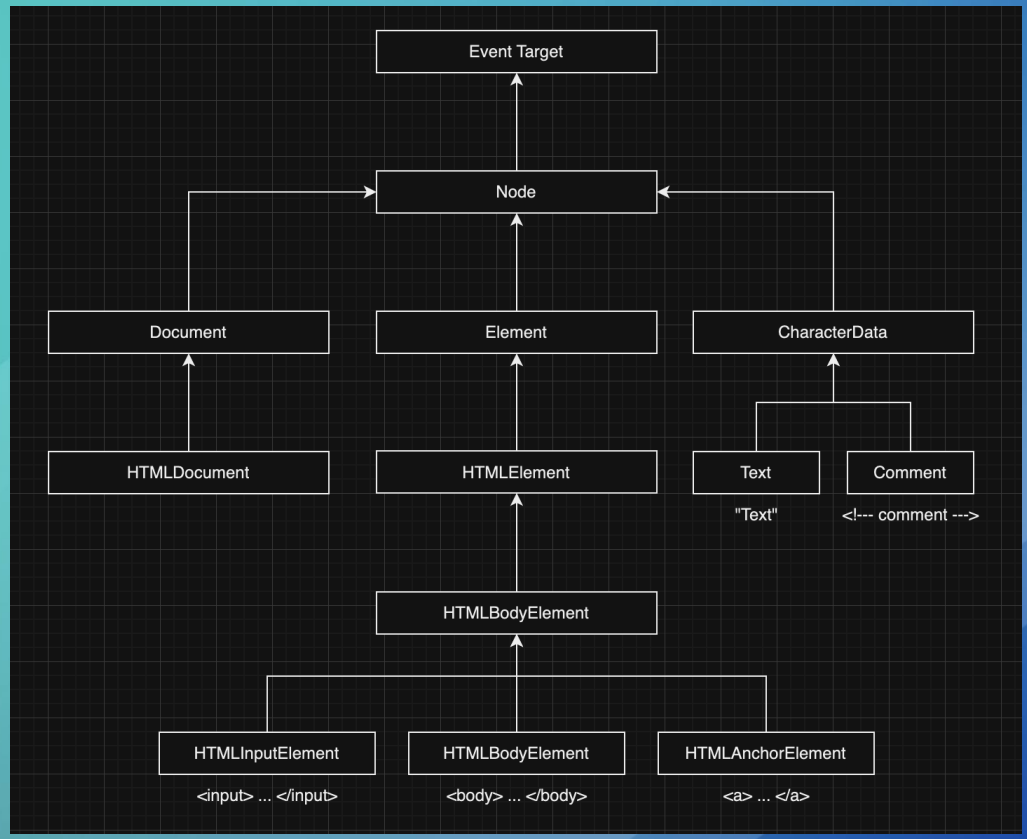
Властивості вузлів

Кожен вузол DOM належить відповідному вбудованому класу.

Коренем ієрархії є EventTarget, від нього успадковується Node, а інші вузли DOM успадкують вже від нього.

- EventTarget – це кореневий “абстрактний” клас для всього.
- Node – також “абстрактний” клас, який є базовим для вузлів DOM. Він надає основні функціональні можливості дерева: parentNode, nextSibling, childNodes та інші геттери.
- Document, з історичних причин його часто успадковує HTMLDocument, представляє собою документ в цілому
- CharacterData – “абстрактний” клас для:

1. Text – клас, що відповідає тексту всередині елементів.
2. Comment – клас для коментарів. Кожен коментар стає частиною DOM.



Властивості вузлів

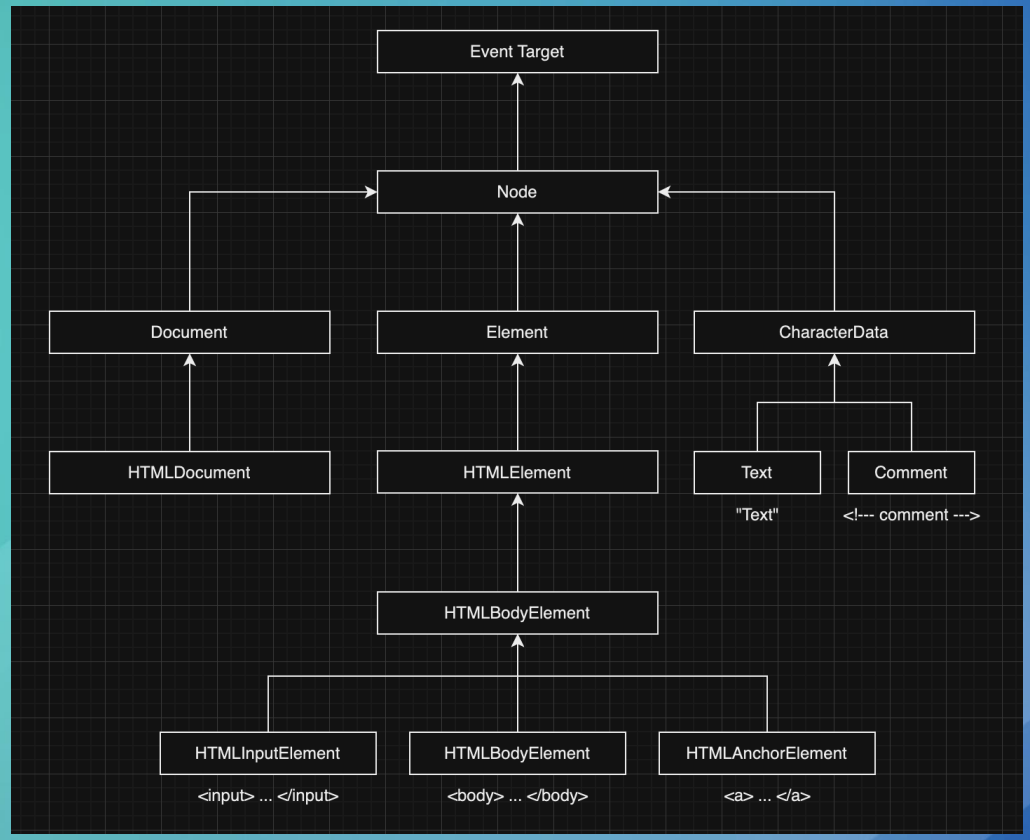
- Element – базовий клас для елементів DOM.

Він надає навігаційні можливості на рівні елементів, такі як nextElementSibling, children та методи пошуку, такі як getElementsByTagName, querySelector.

- HTMLElement – це базовий клас для всіх HTML-елементів.

Він успадковується класами конкретних HTML-елементів:

- HTMLInputElement – клас для елементів <input>,
- HTMLBodyElement – клас для елементів <body>,
- HTMLAnchorElement – клас для елементів <a>,
- ...



nodeType

Властивість nodeType надає спосіб отримання “типу” вузла DOM.

Вона має числове значення від 1 до 12:

- elem.nodeType == 1 для вузлів-елементів,
- elem.nodeType == 3 для текстових вузлів,
- elem.nodeType == 9 для об’єкта документа,

В специфікації можна переглянути всі значення.

nodeName та tagName

Для визначення імені тегу DOM-вузла ми можемо скористатися властивостями nodeName або tagName.

Властивість tagName існує лише для вузлів типу Element.

Властивість nodeName визначається для будь-якого вузла типу Node:

- для елементів вона має те ж значення, що і tagName.
- для інших типів вузлів (текст, коментар тощо) вона містить рядок з типом вузла.

Назва тегів завжди написана великими літерами, за винятком режиму XML

innerHTML

Властивість інтерфейсу Element innerHTML встановлює або отримує HTML або XML розмітку дочірніх елементів.

Використання InnerHTML для вставлення тексту до веб-сторінки. Це призводить до ризиків безпеки.

Якщо innerHTML вставляє тег <script> у документ – він стає частиною HTML, але не виконується.

Однак, є способи запустити JavaScript без використання елементів <script>, так що є ризик безпеки щоразу, коли ви використовуєте innerHTML для набору рядків, над якими у вас немає контролю.

nodeValue || data

Інші типи вузлів, такі як текстові вузли, мають свій аналог: nodeValue і data властивості. Ці дві властивості практично ідентичні за своїм призначенням, є лише незначні різниці в специфікації.

textContent

Властивість textContent надає доступ до тексту всередині елемента: лише текст, без усіх <тегів>. Запис в textContent набагато корисніше, тому що це дозволяє записати текст “безпечним способом”.

hidden

Атрибут “hidden” та властивість DOM визначає видно елемент чи ні.

Технічно, hidden працює так само, як style="display:none". Але це коротше писати.

Атрибути та властивості

Коли браузер завантажує сторінку, він “читає” (іншими словами: “парсить”) HTML і генерує DOM об’єкти з нього. Для вузлів-елементів більшість стандартних атрибутів HTML автоматично стають властивостями об’єктів DOM.

Властивості та методи DOM поводяться так само, як і звичайні об’єкти JavaScript:

- Вони можуть мати будь-яке значення.
- Вони чутливі до регістру (наприклад h1.showName, не h1.ShowName).

У HTML, теги можуть мати атрибути. Коли браузер аналізує HTML і створює DOM-об’єкти для тегів, він розпізнає стандартні атрибути та створює з них властивості DOM.

Отже, коли елемент має id або інший стандартний атрибут, створюється відповідна властивість. Проте цього не відбувається, якщо атрибут не є стандартним.

Атрибути

Всі атрибути доступні за допомогою наступних методів:

- elem.hasAttribute(name) – перевіряє наявність атрибута.
- elem.getAttribute(name) – отримує значення атрибута.
- elem.setAttribute(name, value) – встановлює значення атрибута.
- elem.removeAttribute(name) – видаляє атрибут.
- elem.attributes – це колекція об’єктів, які належать вбудованому класу Attr, і мають властивості name та value.

Нестандартні атрибути data

Щоб уникнути конфліктів, існують data-* атрибути.

Всі атрибути, які починаються з “data-” зарезервовані для використання програмістами. Вони доступні у властивості dataset.

Краще використовувати атрібути ніж класи, тому що це більш гнучкий спосіб.

Атрибути, що складаються з декількох слів, такі як data-user-name стають записані за допомогою верблюжої нотації (camel-case): dataset.userName

Використання атрибутів data-* є валідним, безпечним способом передачі додаткових даних.

Створення та вставка елементу

- document.createElement(tag) - Створює новий елемент з заданим тегом
- document.createTextNode(text) - Створює новий текстовий вузол з заданим текстом

Методи вставки, вказують куди саме буде вставлено вміст:

- node.append(...вузли або рядки) – додає вузли або рядки в кінець node,
- node.prepend(...вузли або рядки) – вставляє вузли або рядки на початку node,
- node.before(...вузли або рядки) – вставляє вузли або рядки попереду node,
- node.after(...вузли або рядки) – вставляє вузли або рядки після node,
- node.replaceWith(...вузли або рядки) – замінює node заданими вузлами або рядками.

insertAdjacentHTML/Text/Element

Універсальний метод: elem.insertAdjacentHTML(where, html).

Where це кодове слово, яке вказує куди вставляти відносно elem. Його значення має бути одним з наступних:

- "beforebegin" – вставити html безпосередньо перед elem,
- "afterbegin" – вставити html в elem, на початку,
- "beforeend" – вставити html в elem, в кінці,
- "afterend" – вставити html безпосередньо після elem.

html є рядок HTML, який вставляється “як HTML”.

Всі методи вставки автоматично видаляють вузол з попереднього місця.

Видалення вузлів

Щоб видалити вузол використовуйте метод node.remove().

cloneNode

Виклик elem.cloneNode(true) створює «глибоку» копію елемента – з усіма атрибутами та піделементами. Якщо ми викличемо elem.cloneNode(false), тоді буде створена копія без дочірніх елементів.

DocumentFragment

DocumentFragment це спеціальний DOM-вузол який служить обгорткою для передачі списку вузлів.

Ми можемо додавати до нього інші вузли, але коли ми вставляємо його кудись, він “зникає”, а замість нього вставляється лише його вміст (контент).

Що воно дає:

1. Зберігання шматків HTML, що не мають спільного предка;
2. Поліпшення продуктивності у разі множинних вставок;

Існує одне місце, де можна зустріти DocumentFragment, не створюючи його через JS. Це властивість вмісту елемента template.

Тег <template> придуманий спеціально для того, щоб зберігати шматки HTML-коду, але раніше не навантажувати ними браузер. Те, що знаходиться всередині цього тега, не є частиною активного DOM-дерева.

Застарілі методи вставки/видалення

- parentElem.appendChild(node)
- parentElem.insertBefore(node, nextSibling)
- parentElem.removeChild(node)
- document.write

Виклик document.write(html) записує html на сторінку “прямо тут і зараз”. Рядок html може бути згенерований динамічно, тож метод досить гнучкий. За допомогою JavaScript ми можемо створити повноцінну вебсторінку та записати її вміст в документ. Виклик document.write працює лише під час завантаження сторінки.

className та classList

elem.className - створює новий CSS клас.

elem.classList – це спеціальний об’єкт, який містить методи для додавання, видалення або перемикання окремого класу.

Методи classList:

- elem.classList.add/remove("class") – додати/видалити клас.
- elem.classList.toggle("class") – додає клас, якщо він не існує, інакше видаляє його.
- elem.classList.contains("class") – перевіряє, чи існує переданий клас, відповідно повертає true/false.

Стилі елемента

Властивість elem.style – це об’єкт, вміст якого відповідає тому, що записано в атрибуті "style". Встановлення elem.style.width="100px" працює точнісінько так само, як рядок width:100px записаний в атрибут style.

Для властивостей, які називаються кількома словами, використовується верблюдячий регістр (camelCase)

Властивості з браузерними префіксами – наприклад, -moz-border-radius, -webkit-border-radius також підпорядковуються цьому правилу: дефіс означає верхній регістр.

Існує спеціальна властивість style.cssText, яка дає змогу встановлювати повний стиль елемента як рядок

Не забувайте додавати одиниці вимірювання CSS до значень.

Властивість style працює лише зі значенням атрибута "style", без врахування CSS-каскаду.

Дякую за увагу