📘 09. Element, Text, Comment, Attribute – 각각의 노드는 무슨 역할을 할까?

1. 눈에 보이지 않는 진짜 구조: DOM의 내부 세계

우리가 앞에서 배운 것처럼, HTML 문서는 단순히 눈에 보이는 텍스트와 태그로만 이루어져 있는 것이 아닙니다.

브라우저는 이 HTML을 읽고, 각 요소들을 **객체처럼 조작할 수 있는 구조(DOM)** 로 변환해줍니다.

→ 이 과정을 통해 생성되는 것이 바로 DOM, 즉 **Document Object Model**입니다.

→ 문서를 객체처럼 만들고, **트리 구조로 연결된 노드들의 집합**으로 재구성하는 것입니다.

---

2. DOM을 구성하는 다양한 '노드'들의 정체

DOM 트리는 마치 **가계도**나 **조직도**처럼 부모-자식 관계로 구성된 트리 구조입니다.

그리고 이 트리를 구성하는 각각의 조각을 우리는 **노드(Node)**라고 부릅니다.

→ 단순히 `<div>` 같은 태그만 있는 게 아닙니다.

→ **텍스트, 주석, 속성도 모두 다른 종류의 노드**입니다.

---

3. 하나의 HTML 태그가 여러 노드로 나뉜다

예를 들어 다음 HTML을 살펴보겠습니다:

```
<div id="box">Hello <span>World</span><!-- 주석입니다 --></div>
```

우리는 이 구조를 하나의 `<div>` 안에 텍스트와 태그, 주석이 있는 단순한 형태로 보지만,

브라우저는 이걸 다음과 같은 **세부적인 노드들로 분해**합니다:

```
Element Node: <div id="box">
├── Text Node: "Hello "
├── Element Node: <span>
│   └── Text Node: "World"
└── Comment Node: " 주석입니다 "
```

→ `Hello`는 Text 노드, `<span>`은 또 하나의 Element 노드,

→ `"주석입니다"`는 눈에 보이지 않지만 Comment 노드로 존재합니다.

→ 이 세 가지는 모두 `<div>`의 **자식 노드**이며, 서로 **형제 노드**이기도 합니다.

---

4. DOM의 4가지 주요 노드 타입

이제 DOM에서 가장 핵심이 되는 4가지 노드 타입을 정리해보겠습니다:

종류	설명
Element Node	`<div>`, `<p>`, `<span>` 같은 HTML 태그들
Text Node	태그 내부의 텍스트, 예: `"Hello"`
Comment Node	`<!-- 주석입니다 -->`와 같은 주석
Attribute Node	태그의 속성, 예: `id="box"`

→ 이 네 가지는 **모두 노드이지만**, 역할과 위치, 접근 방식이 서로 다릅니다.

---

5. `.nodeType`으로 노드 종류를 구분하기

DOM은 각 노드에 고유한 숫자 타입을 부여합니다.

자바스크립트에서는 `.nodeType` 속성을 통해 이 숫자를 확인할 수 있습니다.

노드 종류	`.nodeType` 값
Element Node	`1`
Attribute Node	`2`
Text Node	`3`
Comment Node	`8`

실제로 이를 확인해보면 다음과 같습니다:

```
const div = document.getElementById("box");

console.log(div.nodeType);              // 1: Element
console.log(div.childNodes);            // NodeList(3) [Text, span, Comment]
console.log(div.childNodes[0].nodeType); // 3: Text
console.log(div.childNodes[1].nodeType); // 1: Element
console.log(div.childNodes[2].nodeType); // 8: Comment
```

→ 하나의 `<div>` 안에 다양한 **노드 타입들이 공존**한다는 점을 꼭 기억해야 합니다.

---

6. Attribute 노드는 childNodes에 없다?

하나 주의할 점은, 다음 코드에서의 `"id"` 속성입니다:

```
<div id="box">...</div>
```

→ 이 `id="box"`는 속성이지만, **childNodes 목록에는 포함되지 않습니다.**

→ 속성은 별도의 전용 저장소인 `.attributes`에서 관리됩니다:

```
console.log(div.attributes);            // NamedNodeMap {0: id, id: "box", length: 1}
console.log(div.attributes[0].name);    // "id"
console.log(div.attributes[0].value);   // "box"
```

→ `.attributes`는 배열처럼 보이지만 **NamedNodeMap**이라는 특수한 객체입니다.

→ 실무에서는 대부분 `.getAttribute()`와 `.setAttribute()`로 속성을 다룹니다:

```
div.getAttribute("id");    // "box"
div.setAttribute("id", "newBox");
```

---

7. 비유로 이해하는 4가지 노드

각 노드를 현실에 비유하면 다음과 같습니다:

노드 종류	비유
Element	건물의 뼈대
Text	건물 안에 적힌 문구나 설명
Comment	벽에 붙은 메모, 개발자 참고용
Attribute	문 옆에 붙은 표지판

→ 이 비유를 통해 각 노드의 역할을 시각적으로 떠올리기 쉬워집니다.

---

8. 실전 예제로 확인해보기

아래 HTML 구조가 있다면:

```
<div id="box">Hello <span>World</span><!-- 주석입니다 --></div>
```

다음과 같이 DOM 구조를 콘솔에서 확인할 수 있습니다:

```
const box = document.getElementById("box");

console.log(box.childNodes);        // [Text, span, Comment]
console.log(box.attributes);        // NamedNodeMap {0: id, id: "box"}
console.log(box.getAttribute("id")); // "box"
console.dir(box);                   // 객체 내부 구조 시각화
```

→ `console.dir()`은 `box` 객체의 속성을 **트리처럼 펼쳐 보여주는 디버깅 도구**입니다.

→ 구조를 확인할 때 매우 유용하니 자주 활용하세요.

✅ 핵심 요약

DOM은 단순히 태그의 나열이 아닙니다. **여러 종류의 노드로 구성된 트리 구조**입니다.

- **Element**: 실제 태그 (`<div>`, `<span>`)
- **Text**: 글자 `"Hello"`, `"World"`
- **Comment**: 주석 `<!-- 설명 -->`
- **Attribute**: 태그의 속성 `id="box"`

→ `.childNodes`에는 Text, Element, Comment는 포함되지만

→ **Attribute는 `.attributes`에서 별도로 관리**됩니다.

→ `.nodeType`을 사용하면 어떤 노드인지 숫자로 식별 가능하며,

→ DOM을 정확히 조작하려면 이 차이를 반드시 알아야 합니다.
