📘 07. document 객체는 어떻게 만들어지고 무슨 일을 할까?

1. 왜 항상 `document`로 시작할까?

여러분 혹시 이런 경험 있으신가요?

HTML로 웹 페이지를 만들었는데, 자바스크립트로 어떤 요소를 바꾸려고 할 때마다 `document.querySelector()` 같은 걸 써야 했던 적.

"왜 자꾸 document로 시작하지?" 하고 궁금했던 적이 있다면, 오늘 그 이유를 아주 명확하게 풀어보겠습니다.

→ 우리는 지난 시간에 DOM이 단순한 HTML이 아니라, **객체로 표현된 트리 구조**라는 것을 배웠습니다.

→ 그리고 이 구조 전체의 가장 꼭대기에 있는 것이 바로 `document` **객체**입니다.

---

2. 객체 트리의 시작점: `document`

우선 '객체'라는 말부터 다시 떠올려볼까요?

객체는 이름과 값을 가진 덩어리 데이터입니다. 예를 들어 '사람'이라는 객체는 이름, 나이, 키 같은 정보를 포함할 수 있죠.

→ 마찬가지로 브라우저는 HTML 코드를 단순히 문자로 읽는 것이 아니라, 그걸 **분석해서 객체로 변환**한 뒤, 트리 구조로 화면에 보여줍니다.

그리고 그 트리 구조의 가장 상단, 즉 전체 구조의 시작점이 되는 것이 바로 `document`입니다.

이 `document`는 **웹 페이지 전체의 조립 설명서**와 같은 역할을 합니다.

---

3. 레고 설명서 비유로 이해하는 `document`

예를 들어봅시다. 여러분이 레고 블록을 받았다고 가정해보세요.

처음에는 설명서도 없고, 무엇을 만들지도 모르겠죠. 하지만 레고 회사에서 만든 **조립 설명서**를 보면,

→ 어떤 블록이 어디에 있어야 하는지

→ 어떤 블록이 다른 블록 안에 포함되어야 하는지

→ 전체 구조가 어떻게 조립되어야 하는지

한눈에 이해할 수 있습니다.

→ 이 설명서가 바로 `document` **객체**입니다.

→ HTML 전체 구조를 담고 있는, 브라우저가 자동 생성한 **DOM 트리의 지도**인 셈이죠.

---

4. 브라우저는 HTML을 어떻게 트리로 바꿀까?

아래의 HTML을 살펴보겠습니다:

```
<!DOCTYPE html>
<html>
  <head>
    <title>나의 첫 웹 페이지</title>
  </head>
  <body>
    <h1>안녕하세요!</h1>
  </body>
</html>
```

브라우저는 이 HTML을 열자마자 다음과 같은 **트리 구조**로 해석합니다:

```
Document
└── html
    ├── head
    │   └── title → "나의 첫 웹 페이지"
    └── body
        └── h1 → "안녕하세요!"
```

→ 이 구조는 **메모리 상에 자바스크립트 객체로 변환되어 저장**됩니다.

→ 그리고 이 전체 구조를 가리키는 객체가 바로 `document`입니다.

---

5. 그래서 항상 `document.querySelector()`로 시작한다

우리가 요소를 찾을 때 이렇게 쓰는 이유는 바로 여기에 있습니다:

```
document.querySelector("h1");
```

→ 왜 `querySelector` 앞에 꼭 `document`가 붙어야 할까요?

→ 그 이유는 **`document`가 없으면 DOM 트리 자체를 참조할 수 없기 때문**입니다.

→ 마치 도서관에서 책을 찾을 때, 전체 책 목록이 담긴 관리 시스템(document)을 먼저 참조하는 것과 같습니다.

→ `document`는 DOM 트리 전체의 **관리자이자 출발점**입니다.

---

6. 브라우저가 `document`를 자동으로 만들어준다

그렇다면 이 `document`는 누가 만들까요?

→ 우리가 직접 정의한 적은 없지만, 브라우저는 HTML을 해석하는 순간 **자동으로 DOM 트리를 만들고**,

→ 그 트리를 대표하는 `document` 객체를 **전역(global) 변수로 등록**해줍니다.

→ 그래서 우리는 별도 선언 없이 `console.log(document)`라고 입력할 수 있는 것입니다.

---

7. console.log vs console.dir: 내부 구조 확인하기

```
console.log(document);
```

→ 이 명령어는 DOM 구조를 마치 HTML처럼 렌더링된 상태로 보여줍니다.

하지만 진짜 내부 객체 구조를 보려면 다음을 사용해야 합니다:

```
console.dir(document);
```

→ 이 명령어는 `document` 객체를 **트리 구조의 속성과 메서드까지 포함된 자바스크립트 객체**로 보여줍니다.

→ 어떤 속성이 있고, 어떤 메서드를 사용할 수 있는지 개발자 관점에서 상세하게 확인할 수 있습니다.

---

8. DOM 조작의 출발점이 되는 `document`

이 `document` 객체는 단순한 구조의 시작점일 뿐 아니라, DOM 조작의 **출발점이자 관리자** 역할을 합니다.

```
// 요소 찾기
const title = document.querySelector("h1");

// 요소 만들기
const newDiv = document.createElement("div");

// 값 읽기
const formValue = document.querySelector("input").value;
```

→ 이 모든 코드의 공통점은, **`document`에서부터 시작한다는 것**입니다.

→ DOM을 **찾고, 만들고, 읽고, 지우는 모든 작업**이 document에서 출발합니다.

---

9. document는 HTMLDocument 클래스의 인스턴스다

조금 더 기술적으로 설명하자면, `document`는 실제로는 `HTMLDocument`라는 클래스의 `인스턴스(instance)`입니다.

→ 클래스는 일종의 설계도이고, 인스턴스는 그 설계도로 만든 **실제 객체**입니다.

→ 즉, `document`는 HTML 문서 처리를 위해 브라우저가 만든 **특수한 DOM 객체**입니다.

확인해볼까요?

```
document instanceof HTMLDocument; // true
```

→ 이 결과는 `document`가 HTML 문서를 위한 특별한 객체라는 증거입니다.

✅ 핵심 요약

정리하자면, `document`는 **웹 페이지 전체 DOM 트리를 대표하는 객체**입니다.

- 브라우저가 HTML을 해석하며 자동으로 생성하며,
- 자바스크립트에서 DOM 요소를 **찾고, 만들고, 지우는 모든 동작의 시작점**이 됩니다.

- 도서관 관리 시스템처럼, **모든 정보와 조작의 통로** 역할을 합니다.

→ `document`는 DOM을 다루기 위한 **출발점이자 관리자 객체**입니다.

→ DOM 조작은 항상 이 객체에서 시작되므로, 반드시 제대로 이해하고 넘어가야 합니다.
