13. 📘 querySelector – CSS 문법으로 단일 요소를 잡아낸다

1. CSS처럼 요소를 선택할 수 있다면 얼마나 편할까?

지금까지 우리는 DOM 요소를 선택할 때 `getElementById()`, `getElementsByClassName()`처럼 **메서드 이름에 기준이 명확히 드러나는 방식**들을 사용해왔습니다.

하지만 실제 웹 개발을 하다 보면 이런 상황이 자주 발생합니다:

- "아이디가 main이면서 그 안에 있는 특정 클래스를 가진 요소를 선택하고 싶어"
- "이 섹션 안에 있는 첫 번째 링크만 찾고 싶어"
- "버튼 중에서 `.highlighted` 클래스를 가진 첫 번째 것만 선택하고 싶어"

이럴 때마다 `id`, `class`, `tag`를 따로따로 다루는 방식은 매우 불편하죠.

→ 바로 이런 복잡한 상황을 해결하기 위해 등장한 것이 `querySelector()`입니다.

→ **CSS 선택자 문법을 그대로 사용해서, 가장 먼저 일치하는 요소 하나를 찾는 메서드**입니다.

---

2. 함수 이름을 직역해보면 더 쉬워진다

`querySelector`라는 이름은 처음에는 조금 낯설 수 있지만, 뜻을 그대로 해석해보면 매우 명확해집니다:

- **query** → 찾다, 질의하다
- **selector** → 선택자 (CSS에서 쓰던 선택자 문법)

즉, **"CSS 선택자를 사용해서 요소를 찾는 함수"** 라는 뜻입니다.

---

3. 기본 사용법 – CSS 선택자 그대로 쓰면 된다

예를 들어 클래스가 `btn`인 요소를 찾고 싶다면 아래와 같이 씁니다:

```
const button = document.querySelector(".btn");
```

→ HTML 예시:

```
<button class="btn">제출</button>
```

`.btn`은 CSS 문법에서 클래스를 가리킬 때 사용하는 방식 그대로입니다.

→ 클래스명 앞에 점(.)을 붙여 사용합니다.

---

4. 아이디나 태그도 모두 지원한다

- 아이디로 찾고 싶다면 `#`을 붙입니다:

```
const mainSection = document.querySelector("#main");
```

→ HTML 예시:

```
<div id="main">
  <p>메인 영역입니다.</p>
</div>
```

- 태그 이름으로도 가능합니다:

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

→ HTML 예시:

```
<h1>DOM 완전 정복</h1>
<h1>또 다른 제목</h1>
```

→ 이 경우 가장 위에 있는 첫 번째 `<h1>` 요소 하나만 선택됩니다.

---

5. CSS 선택자 조합도 가능하다 – 진짜 파워는 여기서 시작된다

`querySelector()`의 진짜 강력한 기능은 **복잡한 선택자 조합**입니다.

```
const item = document.querySelector("#container .highlighted");
```

→ 위 코드는 `id="container"` 요소 내부에 있는 `class="highlighted"` 요소를 찾습니다.

→ HTML 예시:

```
<div id="container">
  <p class="highlighted">강조된 텍스트</p>
  <p>일반 텍스트</p>
</div>
```

→ `getElementById()`나 `getElementsByClassName()`으로는 이렇게 한 줄로 조합된 조건을 표현하기 어렵습니다.

---

6. 특정 요소 내부에서만 검색하기

전체 문서가 아니라 **특정 DOM 요소 안에서만** 검색을 제한할 수도 있습니다:

```
const section = document.getElementById("menu");
const firstLink = section.querySelector("a");
```

→ HTML 예시:

```
<div id="menu">
  <a href="#">첫 번째 링크</a>
  <a href="#">두 번째 링크</a>
</div>
```

→ 이 코드는 `"menu"` 섹션 내부의 첫 번째 `<a>` 태그 하나만 선택합니다.

→ 마치 "서랍 안에 있는 연필만 찾아줘!"라고 말하는 것과 같습니다.

---

7. 실습 – 직접 콘솔에서 확인해보기

실제로 어떤 요소가 선택되었는지 확인하고 싶다면 다음처럼 콘솔에서 확인합니다:

```
console.dir(document.querySelector(".btn"));
```

→ `.dir()`은 선택한 요소의 속성 구조를 객체 트리 형태로 자세히 보여주는 도구입니다.

→ HTML과 CSS 구조를 눈으로 확인하며 CSS 선택자 감각도 함께 익힐 수 있습니다.

---

8. 주의사항 – 없으면 `null`, 그래서 조건문이 필요하다

`querySelector()`는 **가장 먼저 일치하는 요소 하나만** 반환합니다.

→ 그런데 일치하는 요소가 없다면? 반환값은 `null`입니다.

즉, 이런 코드는 에러를 일으킬 수 있습니다:

```
document.querySelector(".none").style.color = "blue"; // ❌ TypeError 발생
```

→ 해결 방법: 반드시 조건문으로 존재 여부를 먼저 확인하세요.

```
const result = document.querySelector(".something");
if (result !== null) {
  result.style.color = "blue";
}
```

→ 안전한 DOM 조작을 위해 이 패턴을 습관화하세요.

✅ 핵심 요약

- `querySelector()`는 **CSS 선택자 문법**을 그대로 사용할 수 있는 DOM 요소 선택 메서드입니다.
- 문서 전체 또는 특정 요소 안에서 **가장 먼저 일치하는 요소 하나**를 반환합니다.
- 클래스 (`.class`), 아이디 (`#id`), 태그 (`tag`), 조합 (`#id .class`) 등 모든 CSS 선택자 사용 가능
- 반환값이 없을 경우 `null`이므로 반드시 존재 여부 확인 후 조작해야 안전합니다.

→ `querySelector()`는 **선택자 기반 DOM 조작의 핵심 출발점**이자, CSS 감각과 자바스크립트를 연결하는 중요한 도구입니다.
