DOM(WebAPI)의 객체 document

웹 페이지가 로드될 때 생성되는 객체로, 해당 페이지의 내용을 나타내며, 웹 페이지의 Document Object Model (DOM)의 진입점입니다. document 객체는 웹 페이지의 모든 요소와 그 속성에 접근하고, 조작할 수 있는 다양한 메소드와 속성을 포함하고 있습니다.

이벤트 처리는 사용자 상호작용(클릭, 키보드 입력 등)에 반응하여 특정 작업을 수행하는 자바스크립트의 중요한 기능입니다. 이벤트 리스너를 사용하여 특정 이벤트를 감지하고, 이에 대한 반응으로 이벤트 핸들러 함수를 실행합니다.

노드 탐색

범주	속성/메소드	설명
노드 탐색	`document.getElementById(id)`	ID를 기반으로 요소를 찾습니다.
노드 탐색	`document.getElementsByTagName(name)`	태그 이름을 기반으로 요소들을 찾습니다.
노드 탐색	`document.getElementsByClassName(name)[0]`	클래스 이름을 기반으로 요소들을 찾습니다.
노드 탐색	`document.querySelector(selector)`	CSS 선택자에 해당하는 첫 번째 요소를 찾습니다.
노드 탐색	`document.querySelectorAll(selector)`	CSS 선택자에 해당하는 모든 요소를 찾습니다.
노드 탐색	`document.documentElement`	문서의 루트 요소(HTML)에 접근합니다.
노드 탐색	`document.body`	`<body>` 요소에 접근합니다.
노드 탐색	`document.head`	`<head>` 요소에 접근합니다.

노드 탐색은 웹 페이지의 DOM(문서 객체 모델)에서 특정 요소를 찾는 과정입니다. 이 과정은 웹 페이지의 구조를 탐색하여 필요한 HTML 요소를 검색하고 선택하는 데 사용됩니다. 예를 들어, 특정 ID, 클래스, 태그 이름 또는 CSS 선택자를 기반으로 요소를 찾을 수 있습니다. 노드 탐색을 통해 개발자는 웹 페이지의 다양한 부분에 접근하고, 이를 조작할 수 있습니다.

문서 정보

범주	속성/메소드	설명
문서 정보	`document.title`	페이지의 제목을 가져오거나 설정합니다.
문서 정보	`document.URL`	페이지의 전체 URL을 반환합니다.
문서 정보	`document.domain`	문서의 도메인을 반환합니다.
문서 정보	`document.referrer`	문서를 참조한 페이지의 URL을 반환합니다.

문서 정보는 웹 페이지에 대한 메타 정보를 제공합니다. 이러한 정보에는 웹 페이지의 제목, URL, 로딩된 문서의 도메인, 현재 페이지를 참조한 이전 페이지의 URL 등이 포함됩니다. 이 정보는 웹 페이지의 상태를 이해하거나, 웹 페이지가 실행되고 있는 컨텍스트에 대한 정보를 제공하는 데 유용합니다.

문서 조작

범주	속성/메소드	설명
문서 조작	`document.createElement(tagName)`	새로운 요소를 생성합니다.
문서 조작	`document.createTextNode(data)`	새로운 텍스트 노드를 생성합니다.
문서 조작	`document.appendChild(child)`, `document.removeChild(child)`	요소를 추가하거나 제거합니다.

문서 조작은 DOM을 동적으로 변경하는 기능을 제공합니다. 이를 통해 개발자는 새로운 요소를 생성하고, 기존 요소에 텍스트 노드를 추가하며, 요소를 추가하거나 제거할 수 있습니다. 이 기능은 웹 페이지의 내용을 프로그래매틱하게 수정하고 사용자와의 상호작용에 따라 동적으로 UI를 업데이트하는 데 중요합니다.

이벤트 처리

범주	속성/메소드	설명
이벤트 처리	`document.addEventListener(type, listener)`	특정 이벤트 유형에 리스너를 추가합니다.
이벤트 처리	`document.removeEventListener(type, listener)`	이벤트 리스너를 제거합니다.
기타	`document.cookie`	쿠키에 접근하거나 설정합니다.
기타	`document.forms`	문서 내의 폼 요소에 접근합니다.
기타	`document.images`	문서 내의 이미지 요소에 접근합니다.

범주	속성/메소드	설명
기타	`document.readyState`	문서 로딩 상태를 나타냅니다 (`loading`, `interactive`, `complete`).

이벤트 처리는 웹 페이지에서 발생하는 다양한 사용자 상호작용이나 브라우저 이벤트를 처리합니다. document 객체를 사용하여 이벤트 리스너를 추가하거나 제거할 수 있습니다. 이를 통해 특정 이벤트가 발생했을 때 실행할 함수를 지정할 수 있습니다. 이벤트 처리는 웹 페이지를 대화형이고 반응적으로 만드는 데 중요한 역할을 합니다.

addEventListener 의 type 종류

- click: 요소를 클릭할 때 발생합니다.
- dblclick: 요소를 더블 클릭할 때 발생합니다.
- mousemove: 마우스 포인터가 요소 위를 지날 때 발생합니다.
- mouseover: 마우스 포인터가 요소 위로 올라갈 때 발생합니다.
- mouseout: 마우스 포인터가 요소에서 벗어날 때 발생합니다.
- mousedown: 사용자가 마우스 버튼을 누를 때 발생합니다.
- mouseup: 사용자가 누른 마우스 버튼을 놓을 때 발생합니다.
- keydown: 사용자가 키보드의 키를 누를 때 발생합니다.
- keyup: 사용자가 누른 키보드의 키를 놓을 때 발생합니다.
- keypress: 사용자가 키를 누르고 있을 때 발생합니다. (더 이상 널리 사용되지 않음)
- change: 입력 필드의 값이 바뀔 때 발생합니다 (예: 텍스트 필드, 체크박스).
- submit: 폼이 제출될 때 발생합니다.
- load: 페이지나 이미지가 완전히 로드되었을 때 발생합니다.
- resize: 브라우저 창의 크기가 변경될 때 발생합니다.
- scroll: 스크롤이 발생할 때 (예: 스크롤바, 마우스휠) 발생합니다.

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>동적 요소 조작 예제</title>
  <style>
    body { font-family: Arial, sans-serif; text-align: center; padding:
20px; }
    button { margin: 5px; padding: 10px; }
  </style>
</head>
<body>
  <h1 id="header">안녕하세요!</h1>
  <button id="change-button">텍스트 변경</button>
  <button id="add-button">항목 추가</button>
  <button id="remove-button">항목 제거</button>
  <ul id="item-list">
      <li>항목 1</li>
      <li>항목 2</li>
      <li>항목 3</li>
  </ul>
  <script>
    let changeButton = document.getElementById('change-button');  // ID를 사
```

```
용하여 요소 찾기
    let addButton = document.getElementById('add-button');
    let removeButton = document.getElementById('remove-button');
    let header = document.getElementById('header');
    let itemList = document.getElementById('item-list');

    let items = document.getElementsByClassName('item'); // 클래스 이름을 사용하
여 요소 찾기
    let divs = document.getElementsByTagName('div');     // 태그 이름을 사용하여
요소 찾기
    let firstButton = document.querySelector('button');  // CSS 선택자를 사용
하여 첫 번째 요소 찾기
    let buttons = document.querySelectorAll('button');   // CSS 선택자를 사용
하여 모든 요소 찾기

    changeButton.addEventListener('click', function() {
      header.textContent = '반갑습니다!';
    });

    addButton.addEventListener('click', function() {
      let newItem = document.createElement('li');
      newItem.textContent = '새 항목';
      newItem.className = 'item';     // 새 항목에 'item' 클래스 추가
      itemList.appendChild(newItem);
    });

    removeButton.addEventListener('click', function() {
        if (itemList.children.length > 0) {
            itemList.removeChild(itemList.lastChild);
        }
    });

    console.log(header, items, divs, firstButton, buttons);
  </script>
</body>
</html>
```