DOM 프로퍼티, 컬렉션, 메소드

pdf ready html ready

프로퍼티들

```
<div id="main" lang=ko>
  <span style="color:red">
    안녕
  </span>
</div>
```

```
var element = document.getElementById('main');
```

프로퍼티	설명	예시
id	엘리먼트의 id 값	`element.id // main` id는 중복이 없게 지정해야 한다.
lang	엘리먼트의 언어	`element.lang // ko` 언어이다
style	엘리먼트의 스타일 속성	`element.style //CSSStyleDeclaration` 참고 자료 (영어)
title	엘리먼트의 타이틀	`element.title // ""` 타이틀은 tooltip으로 쓰인다
tagName	엘리먼트의 태그 이름	`element.tagName // DIV` 태그 이름 (Read Only)
innerHTML	엘리먼트 사이의 HTML 텍스트	`element.innerHTML // "<span style ... </span>"`

> 참고 : id는 중복이 없게 설계해야 하지만 중복이 있어도 오류나 경고는 띄우지 않는다.

DOM 트리

```
<ul id="parentul">
    <li id="first">first Child</li>
    <li>Hello World!</li>
    <li id="web">웹프으으으으</li>
    <li>bye!</li>
</ul>
```

```
var parent = document.getElementById("parentul");
var web = document.getElementById("web");
```

DOM 트리 관련 프로퍼티들 (전부 Read Only)

프로퍼티	설명	예시
childElementCount	자식 엘리먼트의 갯수	`parent.childElementCount // 4`
firstElementChild	첫 번째 자식	`parent.firstElementChild //<li id="first" ..</li>`
lastElementChild	마지막 자식	`parent.lastElementChild //<li>bye!</li>`
nextElementSibling	다음 형제 객체	`web.nextElementSibling //<li>bye!</li>`
parentElement	부모 엘리먼트	`web.parentElement //<ul id="parentul" ...`
previousElementSibling	이전 형제 객체	`web.previousElementSibling //<li>Hello World!</li>`

레퍼런스

컬렉션	설명	예시
children	모든 자식 엘리먼트의 배열	`parent.children //[li, li, li, li]` (Read Only)

크기와 위치

프로퍼티	설명
offsetHeight	높이
offsetWidth	너비
offsetLeft	왼쪽 시작점
offsetTop	위쪽 시작점

메소드들

```html
<form action="/submits" id="login">
    <input type="text" id="id"/>
    <input type="password" id="pw"/>
    <input type="submit" value="제출" id="submit" />
</form>
```

```javascript
var loginForm = document.getElementById("login");
var txtId = document.getElementById("id");
var txtPw = document.getElementById("pw");
var btnSubmit = document.getElementById("submit");
var br = document.createElement("br"); // br 태그를 만든다
function method () {
    console.log("clicked!");
}
```

메소드 명	설명	예시
addEventListener()	이벤트 리스너 등록	`txtId.addEventListener('click', method)` click할 때 method를 실행
appendChild()	자식 엘리먼트 추가	`loginForm.appendChild(br)` 마지막에 br 태그를 추가한다.
click()	엘리먼트 클릭	`btnSubmit.click()` 폼이 강제로 제출된다.
focus()	포커스를 가지게 함	`txtId.focus()` 아이디 입력창으로로 포커스가 강제로 옮겨진다.
setAttribute()	객체의 속성지정	`btnSubmit.setAttribute("value","로그인")` 버튼이 로그인으로 바뀐다
insertBefore()	자식 엘리먼트를 특정 엘리먼트 뒤에 추가.	`loginForm.insertBefore(br, txtId)` id 입력창 뒤에 한칸 띄어진다.
querySelector()	지정된 css객체와 일치하는 첫 번째 자식 리턴	`loginForm.querySelector("input[type=password]")` txtPw과 같은 엘리먼트가 선택됨.
removeChild()	자식을 삭제한다	`loginForm.removeChild(btnSubmit)` 제출 버튼이 없어진다.
removeEventListener()	이벤트 리스너를 제거한다	`txtId.removeEventListener('click', method)` click할 때 메소드를 이제 실행하지 않는다.

document 객체 사용

프로퍼티들

앞에 `document.` 은 생략한다.

- `location` : 현재 문서의 URL 정보를 가진 location 객체의 레퍼런스
- `domain` : 서버의 도메인 이름
- `referrer` : 이 문서를 로드한 원래 문서의 URL 문자열. 없으면 빈 문자열.
- `cookie` : 이 문서의 쿠키 값
- `lastModified` : 문서의 최근 수정시간
- `readyState` : 문서의 로딩 상태 loading->interactive->complete 순으로 변한다.
  - loading : 로딩 중
  - interactive : 문서의 해독이 끝난 상태이지만 이미지나 스타일 시트, 프레임은 로딩 중
  - complete : 전부 로딩이 끝남
- `title` : 문서의 제목 문자열 (웹 브라우저에 탭 제목)
- `body` : body객체에 대한 레퍼런스 (body 태그에 대한 DOM 객체)
- `head` : head객체에 대한 레퍼런스 (head 태그에 대한 DOM 객체)
- `defaultView` : 문서에 출력된 브라우저 윈도우에 대한 레퍼런스 (즉 사이트가 실행된 창에 대한 객체)

  > 탭이 하나인 상태에서 document.defaultView.close() 를 하면 윈도우가 종료된다.
  >
  > 여러개이면 안되는 이유가 다른 탭들도 전부 종료되면 안되기 때문이다.
  >
  > document.defaultView는 window와 같다. `document.defaultView == window // true`

- `activeElement` : 문서가 포커스를 받을 때 포커스를 받는 객체 (DOM)
- `documentElement` : html 객체에 대한 레퍼런스 (html 태그에 대한 DOM 객체)
- `URL` : 현재 문서의 URL

컬렉션들

- `images` : 문서 내의 모든 이미지 태그들의 컬렉션

  > `document.images.length` 로 현재 사이트의 이미지의 갯수를 알 수 있음.

- `links` : 문서 내의 모든 링크 (`<a>, <area>`) 들의 컬렉션

- `forms` : 문서 내의 모든 폼 객체들의 컬렉션

메소드들

- `getElementsByTagName(tag)` : 주어진 태그 이름을 가진 모든 태그의 컬렉션

```
document.getElementsByTagName("a")
```

- `getElementsByClassName(class)` : 주어진 class 속성 값을 가진 모든 태그의 컬렉션

```
document.getElementsByClassName('mr-5') // mr-5라는 클래스를 가진 모든 태그
```

- `getElementsByName(name)` : 주어진 name 속성 값을 가진 모든 태그의 컬렉션
- `getElementById(id)` : 주어진 id 속성 값을 가진 첫 번째 DOM 객체 리턴 (중복이 있으면 뒤에 id는 무시)
- `addEventListener(type, method)` : document 객체에 이벤트 리스너를 등록
- `removeEventListener(type, method)` : document 객체에 이벤트 리스너 삭제
- `open()` : document 객체에 등록된 콘텐츠를 모두 지우고, 새로운 HTML 콘텐츠를 쓸 수 있도록 연다.
- `write()` : document에 HTML 콘텐츠를 적는다. DOM트리에 연결되고 브라우저에 출력
- `writeln()` : `write()` 후 \n 추가
- `close()` : document 객체에 있는 HTML 콘텐츠를 브라우저에 출력하고, 더 이상 쓰기를 받지 않음.

> open과 close의 필요성
>
> ```
> myWin = window.open("","popups","width=400,height=300");
> myWin.document.open(); // open은 생략이 가능하다고 한다.
> myWin.document.write("<HTML><HEAD><TITLE>SAMPLE</TITLE></HEAD>");
> myWin.document.write("<BODY>");
> myWin.document.write("<P align=center><b>안녕하세요?</B></P>");
> myWin.document.write("</BODY></HTML>");
> myWin.document.close();
> // close를 하면 document에 write된 내용들을 전부 브라우저에 출력하고 더이상 쓰기를
> ```

- `createElement(tag)` : 해당 tag로 엘리먼트를 만듦

```
var atag = document.createElement("a");
```