DOM

DOM

DOM

DOM

- Document Object Model
- HTML에 있는 태그를 객체화하여 자바스크립트에서 다룰 수 있게 한 것
- 모든 노드객체에 접근할 수 있는 요소와 메소드를 제공
  ※ 노드 : HTML에 있는 태그를 구조화 하였을 때 각각의 태그

DOM

요소노드(elements node)

- 요소노드 : 태그 그 자체를 의미

텍스트노드(textnode)

- 텍스트노드 : 태그에 기록되어 있는 문자

※ 텍스트 노드를 가지는 태그와 가지지 않는 태그가 있음
→ 텍스트노드를 가지는 태그 : h?,p…등등
→ 텍스트노드를 가지지 않는 태그 : img, input… 등등

DOM

텍스트 노드가 있는 문서객체 생성

- 요소노드와 텍스트노드를 생성하고 이를 body노트의 자식으로 포함 가능

메소드	내용
document.createElement(“태그명”)	요소노드 생성
document.createTextNode(“내용”)	텍스트 노드 생성
객체명.appendChild(node)	태그에 자손으로 노드 추가

절차

요소 노드 생성 → 텍스트 노드 생성 → 요소 노드에 텍스트 노트 추가
→ body내부의 필요한 위치에 요소 노드 추가

DOM

텍스트 노드가 없는 문서객체 생성

- 요소노드를 생성하고 속성을 설정한 후 이를 body노드의 자식으로 포함 가능

메소드	내용
객체명.속성 = 속성값	태그 속성값 설정
객체명.setAttribute(속성명,속성값)	태그 속성값 설정
객체명.getAttribute(속성명)	태그 속성값 확인
객체명.appendChild(node)	태그에 자손으로 노드 추가

절차

요소 노드 생성 → 생성된 노드 속성 설정
→ body내부의 필요한 위치에 요소 노드 추가

DOM

문서 객체 스타일 수정

- style객체를 이용하여 문서의 스타일을 변경

방법

객체명.style.속성명 = 속성값;

→ 자바스크립트에서 속성명에 '-'를 사용할 수 없기 때문에 css 속성 중 '-'가 사용되는 속성명의 경우 카멜표기법으로 변경해서 사용해야 함
(ex. background-color → backgroundColor)

DOM

문서 객체 제거

- 페이지에 작성되어 있는 문서의 객체(태그)를 제거하는 것

메소드	내용
객체명.remove();	해당 객체 삭제
doucument.removeChild(객체명);	부모객체 내부의 자손태그 삭제

html 태그 접근

javascript

document.getElementById(“아이디명”)

- 태그의 id 속성의 값을 이용해서 태그 엘리먼트 객체 정보를 가져옴
- id속성은 페이지 내에서 태그의 유일한 식별자 역할
- 리턴은 단일 엘리먼트(중복 id를 여러 개 사용 해도 1개만 리턴)

```
const 변수 = document.getElementById(“아이디명”);

console.log(변수.innerHTML);
```

javascript

document.getElementsByClassName(“클래스명”)

- 태그의 class 속성의 값을 이용해서 태그 엘리먼트 객체 정보를 가져옴
- 동일한 class 속성 값을 가진 엘리먼트들을 모두 가져옴
- 리턴은 엘리먼트 객체 배열(해당 class 속성의 엘리먼트가 1개여도 배열로 리턴)

```
const 변수 = document.getElementByClassName(“클래스명”);

console.log(변수[0].innerHTML);
```

javascript

document.getElementsByName("이름")

- 태그의 name 속성의 값을 이용해서 태그 엘리먼트 객체 정보를 가져옴
- 동일한 name 속성 값을 가진 엘리먼트들을 모두 가져옴
- 리턴은 엘리먼트 객체 배열(해당 name 속성의 엘리먼트가 1개여도 배열로 리턴)

```
const 변수 = document.getElementByName("이름");

console.log(변수[0].innerHTML);
```

javascript

document.getElementsByTagName(“태그명”)

- 태그의 태그명을 이용해서 태그 엘리먼트 객체 정보를 가져옴
- 태그 엘리먼트들을 모두 가져옴
- 리턴은 엘리먼트 객체 배열(해당 태그 엘리먼트가 1개여도 배열로 리턴)

```
const 변수 = document.getElementByTagName(“태그이름”);

console.log(변수[0].innerHTML);
```

javascript

document.querySelector(“선택자”)

- id, class, name, tag를 제한하지 않고 css 선택자를 이용하여 태그 엘리먼트 객체 정보를 가져옴
- 같은 선택자로 여러 태그가 존재하더라도 최상단 엘리먼트 1개정보만 가져옴

const 변수 = document.querySelector(“선택자”);

document.querySelectorAll(“선택자”)

- document.querySelector()와 동일하게 사용하며, 한번에 여러 태그 엘리먼트 객체정보를 배열형태로 가져옴
- 선택자로 선택된 엘리먼트가 1개여도 배열형태로 가져오기 때문에 인덱스번호 사용

const 변수 = document.querySelectorAll(“선택자”);

※ querySelector, querySelectorAll 두 문법 모두 ES6에 추가된 문법으로 오래된 브라우저에서 정상적으로 동작하지 않을 수 있음