Javascript 요약집

Javascript 선택자(1)

document.getElementById('태그의 id 속성 값');

()안에 작성한 id를 갖는 태그를 선택한다.

ex> const tag = document.getElementById('java');

위의 코드는 id 속성 값이 'java'인 태그를 선택한다. Id 속성 특성상 일치하는 id를 지닌 하나의 태그만을 선택한다.

document.getElementByClassName('태그의 class 속성 값');

ex> const tag = document.getElementByClassName('java');

위의 코드는 class 속성 값이 'java'인 태그를 모두 선택한다. Class 속성 특성상 동일한 class명을 지닌 태그가 다수 일 수 있기 때문에 배열로 리턴한다.

document.getElementByTagName('태그명');

ex> const tag = document.getElementByTagName('input');

위의 코드는 모든 input태그를 선택한다. 태그명 또한 class명처럼 같은 태그가 다수 일 수 있기 때문에 배열로 리턴한다.

Javascript 선택자(2)

document.querySelector('선택자');

()안에 작성한 css선택자를 통해 하나의 태그를 선택한다.

만약 선택한 태그가 여러 개라면 가장 처음 선택되는 태그만 선택한다.

ex> document.querySelector('h1') -> 가장 처음 나오는 h1 tag 태그를 선택한다.

document.querySelector('#tag') -> id속성 값이 tag인 태그를 선택한다.

document.querySelector('.tag') -> 가장 처음 나오는 class속성 값이 tag인 태그를 선택한다.

document.querySelector('#tag > div') -> id속성 값이 tag인 태그의 자식 div 태그 중 첫번째 자식만 선택한다.

document.querySelectorAll('선택자');

()안에 작성한 css선택자를 통해 태그들을 선택한다.

선택한 태그가 하나여도 여러 태그를 선택한 것처럼 문법을 사용해야 한다.

ex> document.querySelectorAll('h1') -> h1 태그를 모두 선택한다.

document.querySelectorAll('.tag') -> class속성 값이 tag인 태그를 모두 선택한다.

document.querySelectorAll('#tag > div') -> id속성 값이 tag인 태그의 자식 div 태그 전체를 선택한다.

Javascript 선택자(3)

```
<body>
<div class="div1">
    <div id="div1_1">hello</div>
    <div></div>
    <div></div>
</div>
<div class="div1">
    <div></div>
    <div></div>
    <div></div>
</div>
</body>
```

document.querySelector(‘.div1’);

class 속성값이 ‘div1’인 태그는 두개지만 queySelector는 하나의 태그만 선택할 수 있기 때문에 첫번째로 나오는 class 속성값이 ‘div1’인 태그만 선택한다.

document.querySelectorAll(‘#div1_1’);

id 속성이 ‘div1_1’인 태그는 하나뿐이지만 querySelectorAll의 의미는 모두 선택한다는 의미이기 때문에 사용에 주의해야 한다.

ex>

```
const tag = document.querySelectorAll(‘#div1_1’);
alert(tag.innerText); //오류발생
alert(tag[0].innerText); //hello를 alert으로 띄움.
alert(tag[1].innerText); //오류발생
```

Javascript 선택자(4)

선택한 태그에서 부모, 자식, 형제 노드(태그)로 이동(선택)할 수 있다.

1. 자식 노드(태그) 선택

선택한태그.children -> 선택한 태그의 전체 자식 태그를 선택

선택한태그.children[0] -> 선택한 태그의 전체 자식 중 첫번째 자식 태그를 선택

선택한태그.firstElementChild -> 선택한 태그의 첫 번째 자식 태그 선택

선택한태그.lastElementChild-> 선택한 태그의 마지막 자식 태그 선택

2. 부모 노드(태그) 선택

선택한태그.parentElement -> 선택한 태그의 부모 태그 선택

3. 형제 노드(태그) 선택

선택한태그.nextElementSibling -> 선택한 태그의 다음 형제 태그 선택

선택한태그.previousElementSibling -> 선택한 태그의 이전 형제 태그 선택

Javascript 많이 사용하는 기능(속성 값 읽고 쓰기)

선택한태그.속성명 으로 태그의 속성값을 읽거나 변경 할 수 있다.

아래와 같은 코드가 있을 때

<input id="input1" type="text" class="myInput" value="3">

const tag = document.querySelector('#input1'); -> id 속성값이 input1인 태그를 선택

console.log(tag.type); -> text 출력

console.log(tag.class); -> myInput 출력

console.log(tag.value); -> 3 출력

tag.type = 'number'; -> 선택한 태그의 type 속성값을 number로 변경

tag.value = 5; -> 선택한 태그의 value값을 5로 변경

Javascript 많이 사용하는 기능(css 값 읽고 쓰기)

선택한태그의 css(style)를 읽거나 변경 할 수 있다.

const tag = document.querySelector('#input1'); -> id 속성값이 input1인 태그를 선택

tag.style.width = '300px'; -> 선택한 태그의 width속성을 300px로 변경

tag.style['width'] = '300px'; -> 선택한 태그의 width속성을 300px로 변경

tag.style.backgroundColor = 'blue'; -> 선택한 태그의 배경색을 파랑색으로 변경

tag.style['backgroundColor'] = 'blue'; -> 선택한 태그의 배경색을 파랑색으로 변경

tag.style.display = 'none'; -> 해당 요소 숨기기 (요소가 사라지며 공간도 지워진다)

tag.style.display = 'block'; -> 해당 요소 보이기

tag.style.visibility = 'hidden'; -> 해당 요소 숨기기(요소는 보이지 않지만 공간은 유지)

tag.style.visibility = 'visible'; -> 해당 요소 보이기

Javascript 많이 사용하는 기능(기타)

tag.length -> 선택한 태그의 개수 반환

tag.innerText -> 선택한 태그 내부의 글자를 반환

tag.textContent -> 선택한 태그 내부의 글자를 반환(추천 방법)

tag.innerHTML -> 선택한 태그의 내부 HTML 코드를 반환

tag.submit() -> 선택한 폼 태그를 submit (반드시 폼 태그를 선택해야 함)

tag.classList.add('클래스명') -> 선택한 태그에 클래스를 추가한다.

tag.classList.remove('클래스명') -> 선택한 태그에 클래스를 삭제한다.

tag.classList.toggle('클래스명') -> 선택한 태그에 클래스를 토글한다(있으면 삭제, 없으면 추가)

tag.classList.constains('클래스명') -> 선택한 태그에 클래스가 있는지 확인한다. 클래스가 있다면 true, 없다면 false를 리턴

tag.checked -> 라디오, 체크박스의 체크 여부를 확인한다. 체크가 되어있다면 true, 안되어 있다면 false를 리턴

tag.closest('찾을 태그') -> 현재 태그를 감싸고 있는 가장 가까운 찾을 태그를 찾는다.

Ex> 태그.closest('div') -> 현재 태그를 감싸고 있는 가장 가까운 div 태그를 찾아 감

Javascript 태그(노드) 추가, 삭제

태그(노드) 삭제

tag.remove(); -> 선택한 태그를 삭제한다.

부모태그.removeChild('자식태그'); -> 선택한 부모 태그 안에서 선택한 자식태그를 삭제한다.

부모태그.replaceChildren(); -> 원래 기능은 선택한 부모 태그 안에서 ()안에 들어온 자식 태그를 변경하는 기능이지만,
()안에 아무것도 적지 않으면 선택한 부모 태그 안의 모든 자식 태그를 삭제한다.
부모태그.textContent = ''; -> 마찬가지로 부모태그 안의 모든 내용 삭제

태그(노드) 추가

tag.insertAdjacentHTML('추가할 위치', '추가할 태그')

추가할 위치

beforebegin -> 선택한 태그 바로 전에 추가한다. (이전 형제 노드로 추가된다)

afterbegin -> 선택한 태그가 시작되자 마자 태그를 추가한다. (첫번째 자식 노드로 추가 된다)

beforeend -> 선택한 태그가 끝나기 직전에 태그를 추가한다. (마지막 자식 노드로 추가 된다)

afterend -> 선택한 태그가 끝나면 추가한다. (다음 형제 노드로 추가된다)

Javascript 반복문(for-each문)

const arr = [1, 2, 3, 4, 5]; //자바스크립트의 배열

1. 기본 for문

```
for(const i = 0 ; i< arr.length ; i++){
    console.log(i);
}
```

2. for-each 첫번째

```
for(const i of arr){
    console.log(i);
}
```

3. for-each 두번째

```
arr.map(function(value, index){
  console.log(index + ' / ' + value);
});
```

0 / 1

1 / 2

2 / 3

3 / 4

4 / 5

Javascript 이벤트

이벤트 기본 문법

```
tag.addEventListner('이벤트 발생 시기', function(){
    //이벤트 발생 시 동작할 코드
});
```

이벤트 발생 시기

click -> 해당 태그를 클릭했을 때

dbclick -> 해당 태그를 더블 클릭 했을 때

mousedown -> 마우스 버튼을 누르고 있을 때

mouseup -> 눌렀던 마우스를 뗄 때

mousemove -> 마우스를 움직였을 때

mouseover -> 해당 태그에 마우스가 올라왔을 때

mouseout -> 해당 태그로부터 마우스가 벗어났을 때

keydown -> 키를 눌렀을 때

keypress -> 키를 누르고 있을 때

keyup -> 누른 키를 뗄 때

input -> input, textarea태그의 값이 변경되었을 때

change -> 체크박스, 라디오, select 의 상태가 바뀔 때

load -> 해당 페이지가 로드 되었을 때

resize -> 브라우저 창 크기를 조절했을 때

scroll -> 페이지를 스크롤 할 때