[DOM]

웹 브라우저에 표시되는 문서를 동적으로 제어할 수 있도록 해주는 인터페이스

[DOM 을 사용하는 방법]

제어할 대상을 찾아야 한다.



이런 메소드를 통해서 말이다.

기본적으로 HTML 문서를 만들면 각각의 Tag를 바탕으로 해서 브라우저가 웹페이지를 만든다.

또한 그 과정에서 각각의 Tag에 해당되는 객체들을 만들어 놓는다.

작업을 하려면 브라우저가 만들어 놓은 그 객체를 찾아서 조작해야 한다.

[document.getElementsByTagName()]

위 명령어는 인자 명과 같은 Tag들을 모아서 리스트들을 NodeList라는 배열과 유사한 형식으로 반환한다.

NodeList는 배열은 아니지만, length, 배열 연산자를 통해 컨트롤 할 수 있다.

자세히는 나중에 배울 것이다.

        <ol>

            <li>asdf</li>

            <li>asdf</li>

            <li>asdf</li>

        </ol>

        <ul>

            <li>asdf</li>

            <li>asdf</li>

            <li>asdf</li>

        </ul>

        <script>

            var a = document.getElementsByTagName('ul')[0];

위와 같이 쓸 수 있다.

html에서 ol 밑의 li 태그가 있고, ul 밑에 li 태그가 있다.

여기서 document.getElementsByTagName(‘li’)를 하면 li를 모두 빼올 수 있지만, ul 밑의 li만 뽑고싶다면 NodeList즉 배열의 성질을 이용할 수 있다.

그러고 싶다면, 문서 전체에 해당되는 객체를 뽑지 말고, ul에 해당하는 객체를 찾고, 그 객체에 대해서

객체.getElementsByTagName(‘li’) 만을 조회할 수 있다.

물론 더 효율적인 방법이 있지만, 일단 그렇다고.

Document.get~~이 아니라 임의의 html 객체.get~~이 가능하다는 것이다!

[객체.get]

        <ol>

            <li>asdf</li>

            <li>asdf</li>

            <li>asdf</li>

        </ol>

        <ul>

            <li>asdf</li>

            <li>asdf</li>

            <li>asdf</li>

        </ul>

        <script>

            var a = document.getElementsByTagName('ul')[0];

            var lis = a.getElementsByTagName('li');

다른거 말고 script 내의 코드를 보자.

Lis = document~~ 가 아니다.

a라는 객체.getElement~~ 이다!!!!!!!!!!!!!!!

너무 놀랍다.

a에는 문서에서 ul이라는 tag를 가진 원소를 넣는다. 근데 문서내의 ul은 하나밖에 없다. 그래서 리스트의 원소는 1개밖에 없으므로 인덱스는 0으로밖에 못들어갈 것이다.

그럼 그 객체 내에서 Tagname으로 li를 찾으면? Ul 밑에 있는 li 객체들만 뽑힌다!!!!!!!

[Class name을 기준으로 조회하기]

        <ul>

            <li class="active">asdf</li>

            <li class="active">asdf</li>

            <li>asdf</li>

        </ul>

        <script>

            var lis = document.getElementsByClassName('active');

            for(var i=0; i < lis.length; i++) {

                lis[i].style.color = "red";

            }

요렇게 조회한다!

이미 이건 알고 있었겠지만.

근데 왜 ClassName일까?

왜냐하면 ByClass 라는 말은 마치 javascript의 class를 통해 조회하는 것이라고 말하는 것 같아서.

[Id를 이용해서 조회하기]

왠만한 상황에서 Id로 조회하는 것이 성능이 가장 좋다.

        <ul>

            <li id="active">asdf</li>

            <li class="active">asdf</li>

            <li>asdf</li>

        </ul>

        <script>

            var li = document.getElementById("active");

            li.style.color = "red";

이렇게 쓰는 것이다.

그런데 이 상황에서 document.getElementById(“active”)는 리스트를 반환하지 않는다. 그냥 객체를 반환한다.

따라서 length도 안먹는다.

[querySelector 쓰기]

Css는 선택자라는 것이 있다.

Css는 선택자를 이용해서 꾸며주고자 하는 엘리먼트들을 꾸며줄 수 있었다.

querySelector도 이와 같이 선택자를 이용해서 컨트롤하고자 하는 객체들을 리턴해주는 메서드이다.

        <ul>

            <li class="active">asdf</li>

            <li class="active">asdf</li>

            <li>asdf</li>

        </ul>

        <script>

            var li = document.querySelectorAll('li');

            li.style.color = "red";

위 코드에서, li태그를 선택하는 코드를 짰다.

그런데 li 코드는 여러 개다. querySelector는 수많은 li태그 중 1개만 선택한다.

여러 개를 선택하려면 querySelectorAll 을 써서 리스트로 받아와야 한다.

        <ul>

            <li class="active">asdf</li>

            <li class="active">asdf</li>

            <li>asdf</li>

        </ul>

        <script>

            var li = document.querySelectorAll('li');

            for (var i=0; i < li.length; i++) {

                li[i].style.color = "red";

            }

            var li = document.querySelector('.active');

            li.style.color = "blue";

위와 같이, querySelector는 css 선택자처럼 요소를 가져올 수 있다. 이때 1개만 가져온다

querySelectorAll도 css 선택자처럼 요소를 가져오는데, 여러 개를 또 노드리스트(NodeList) 형식으로 가져온다.

[for in문 자바스크립트서 가능?]

var lis = document.querySelectorAll('ul li');

            for (var i in lis) {

                lis[i].style.color = 'blue';

            }

가능!