JavaScript (Part 2)

Today's topics

- 문서객체 (DOM)
- 브라우저 객체

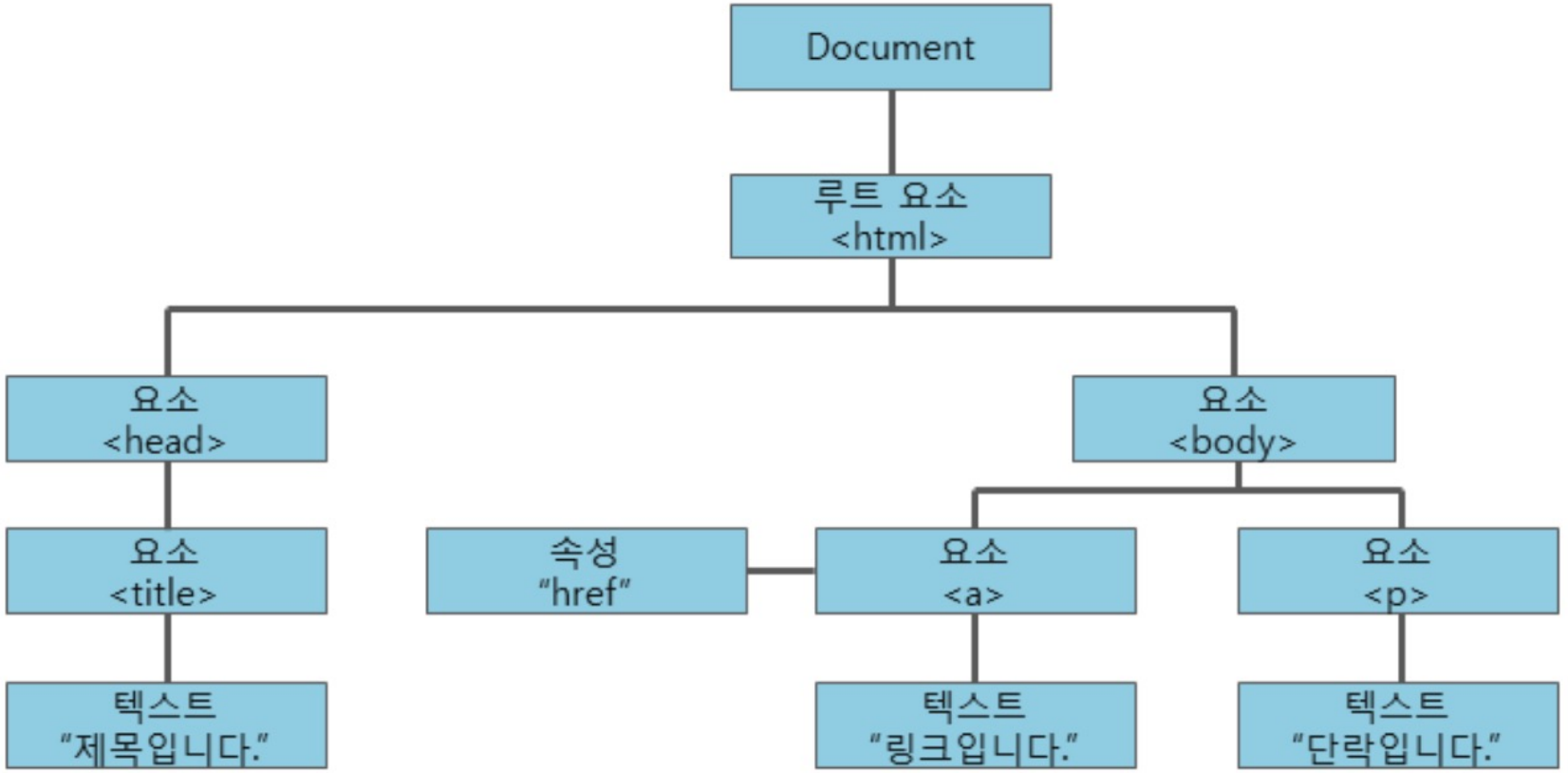
JavaScript 문법 2: 문서객체

-- 튜토리얼 by w3schools & tcpschool.com

문서객체모델 (DOM)

- 문서 객체 모델(DOM, Document Object Model)은 XML이나 HTML 문서에 접근하기 위한 일종의 인터페이스
- 이 객체 모델은 문서 내의 모든 요소를 정의하고, 각각의 요소에 접근하는 방법을 제공

DOM의 계층구조



자바스크립트와 DOM

- 자바스크립트는 새로운 HTML 요소나 속성을 추가
- 자바스크립트는 존재하는 HTML 요소나 속성을 제거
- 자바스크립트는 HTML 문서의 모든 HTML 요소를 변경
- 자바스크립트는 HTML 문서의 모든 HTML 속성을 변경
- 자바스크립트는 HTML 문서의 모든 CSS 스타일을 변경
- 자바스크립트는 HTML 문서에 새로운 HTML 이벤트를 추가
- 자바스크립트는 HTML 문서의 모든 HTML 이벤트에 반응

Document 객체

- Document 객체는 웹 페이지 그 자체
- 웹 페이지에 존재하는 HTML 요소에 접근하고자 할 때는 반드시 Document 객체부터 시작

HTML 요소 선택

- document.getElementsByTagName(태그이름)
  - 태그 이름의 요소를 모두 선택
- document.getElementById(아이디)
  - 아이디의 요소를 선택
- document.getElementsByClassName(클래스이름)
  - 클래스에 속한 요소를 모두 선택
- document.getElementsByName(name속성값)
  - Name속성값을 갖는 요소 모두 선택

HTML 요소 생성

- document.createElement(HTML요소)
  - HTML 요소 생성
- document.write(텍스트)
  - HTML 출력 스트림을 통해 텍스트를 출력

HTML 이벤트 핸들러 추가

- document.getElementById(아이디).onclick = function(){ 실행할 코드 }
  - 마우스 클릭 이벤트와 연결될 이벤트 핸들러 코드를 추가

HTML 객체 선택

document.anchors	name 속성을 가지는 <a>요소를 모두 반환함.
document.body	<body>요소를 반환함.
document.cookie	HTML 문서의 쿠키(cookie)를 반환함.
document.domain	HTML 문서가 위치한 서버의 도메인 네임(domain name)을 반환함.
document.forms	<form>요소를 모두 반환함.
document.images	<img>요소를 모두 반환함.
document.links	href 속성을 가지는 <area>요소와 <a>요소를 모두 반환함.
document.title	<title>요소를 반환함.
document.URL	HTML 문서의 완전한 URL 주소를 반환함.
document.baseURI	HTML 문서의 절대 URI(absolute base URI)를 반환함.

HTML 객체 선택

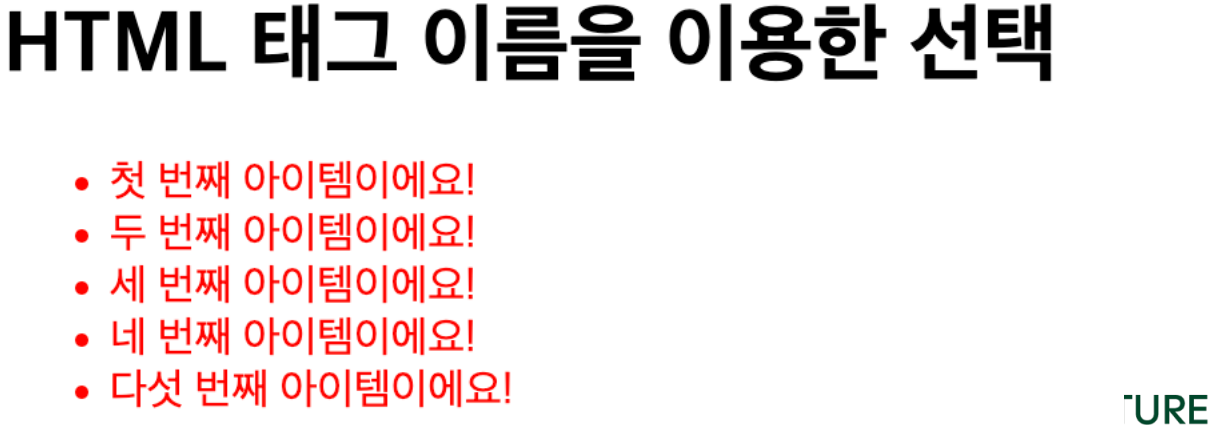
document.doctype	HTML 문서의 문서 타입(doctype)을 반환함.
document.documentElement	<html>요소를 반환함.
document.documentMode	웹 브라우저가 사용하고 있는 모드를 반환함.
document.documentURI	HTML 문서의 URI를 반환함.
document.embeds	<embed>요소를 모두 반환함.
document.head	<head>요소를 반환함.
document.lastModified	HTML 문서의 마지막 갱신 날짜 및 시간을 반환함
document.readyState	HTML 문서의 로딩 상태(loading status)를 반환함.
document.scripts	<script>요소를 모두 반환함.

DOM 요소의 선택

1. HTML 태그 이름(tag name)을 이용한 선택
2. 아이디(id)를 이용한 선택
3. 클래스(class)를 이용한 선택
4. name 속성(attribute)을 이용한 선택
5. CSS 선택자(selector)를 이용한 선택
6. HTML 객체 집합(object collection)을 이용한 선택

1. HTML 태그 이름을 이용한 선택

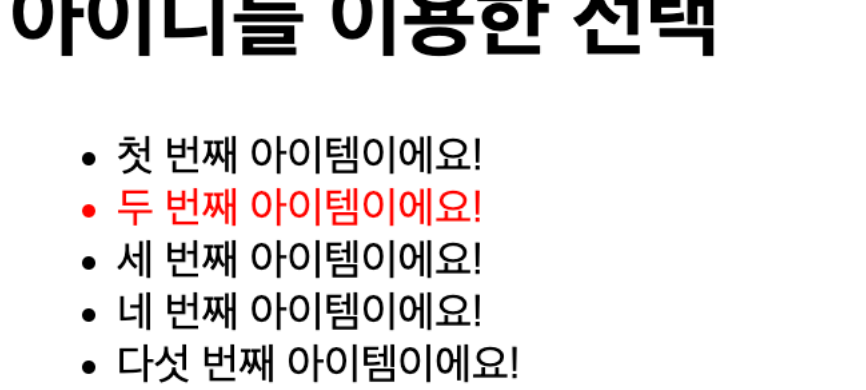
```
<body>
    <h1>HTML 태그 이름을 이용한 선택</h1>
    <ul>
        <li>첫 번째 아이템이에요!</li>
        <li>두 번째 아이템이에요!</li>
        <li>세 번째 아이템이에요!</li>
        <li>네 번째 아이템이에요!</li>
        <li>다섯 번째 아이템이에요!</li>
    </ul>
    <script>
    var selectedItem = document.getElementsByTagName("li"); // 모든 <li> 요소를 선택함.
    for (var i = 0; i < selectedItem.length; i++) {
        selectedItem.item(i).style.color = "red"; // 선택된 모든 요소의 텍스트 색상을 변경함.
    }
    </script></body>
```



2. 아이디를 이용한 선택

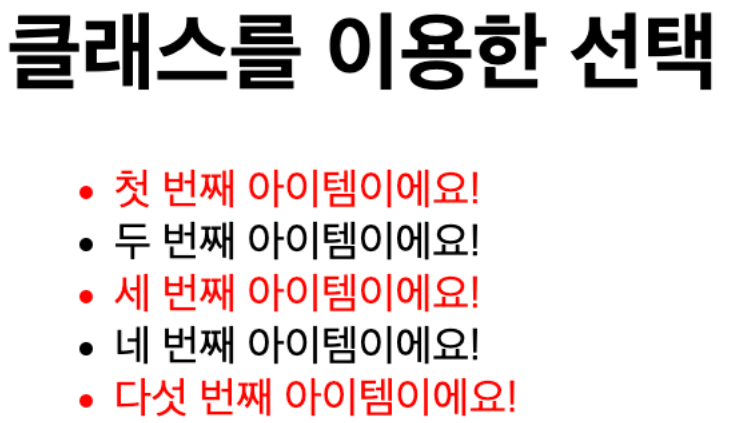
```
<h1>아이디를 이용한 선택</h1>
        <ul>
                <li>첫 번째 아이템이에요!</li>
                <li id="even">두 번째 아이템이에요!</li>
                <li>세 번째 아이템이에요!</li>
                <li id="even">네 번째 아이템이에요!</li>
                <li>다섯 번째 아이템이에요!</li>
        </ul>
```

```
var selectedItem = document.getElementById("even"); // 아이디가 "even" 인 요소를 선택함.
selectedItem.style.color = "red"; // 선택된 요소의 텍스트 색상을 변경함.
```



3. 클래스를 이용한 선택

```
<h1>클래스를 이용한 선택</h1>
        <ul>
                <li class="odd">첫 번째 아이템이에요!</li>
                <li>두 번째 아이템이에요!</li>
                <li class="odd">세 번째 아이템이에요!</li>
                <li>네 번째 아이템이에요!</li>
                <li class="odd">다섯 번째 아이템이에요!</li>
        </ul>

    var selectedItem = document.getElementsByClassName("odd")
    ; // 클래스가 "odd" 인 모든 요소를 선택함.
    for (var i = 0; i < selectedItem.length; i++) {
       selectedItem.item(i).style.color = "red"; // 선택된 모든 요소
    의 텍스트 색상을 변경함.
    }
```



4. Name 속성을 이용한 선택

```
<h1>name 속성을 이용한 선택</h1>
        <p name="first">첫 번째 단락이에요!</p>
        <ul>
                <li name="first">첫 번째 아이템이에요!</li>
                <li>두 번째 아이템이에요!</li>
                <li>세 번째 아이템이에요!</li>
                <li>네 번째 아이템이에요!</li>
                <li>다섯 번째 아이템이에요!</li>
        </ul>

  var selectedItem = document.getElementsByName("first"); // n
  ame 속성값이 "first" 인 모든 요소를 선택함.
  for (var i = 0; i < selectedItem.length; i++) {
     selectedItem.item(i).style.color = "red"; // 선택된 모든 요소
  의 텍스트 색상을 변경함.
  }
```

name 속성을 이용한 선택

첫 번째 단락이에요!

- 첫 번째 아이템이에요!
- 두 번째 아이템이에요!
- 세 번째 아이템이에요!
- 네 번째 아이템이에요!
- 다섯 번째 아이템이에요!

6. HTML 객체 집합을 이용한 선택

```
<html lang="ko">

<head>
        <meta charset="UTF-8">
        <title>JavaScript DOM Element</title>
</head>

<body>

        <h1>HTML 객체 집합을 이용한 선택</h1>

        <script>
                var title = document.title;             // <
title> 요소를 선택함.
                document.write(title);
        </script>

</body>

</html>
```

HTML 객체 집합을 이용한 선택

JavaScript DOM Element

EWHA, THE FUTURE WE CREATE

DOM 요소의 변경

- DOM 요소의 내용변경
  - 출력내용 변경
  - 링크변경
- DOM 요소의 스타일 변경

DOM 요소의 내용변경 1

```
<!DOCTYPE html>
<html lang="ko">

<head>
        <meta charset="UTF-8">
        <title>JavaScript DOM Element</title>
</head>

<body>

        <h1>innerHTML을 이용한 요소의 내용 변경</h1>
        <p id="text">이 문장을 바꿀 것입니다!</p>

        <script>
                var str = document.getElementById("text");
                str.innerHTML = "이 문장으로 바뀌었습니다!";
        </script>

</body>

</html>
```

innerHTML을 이용한 요소의 내용 변경

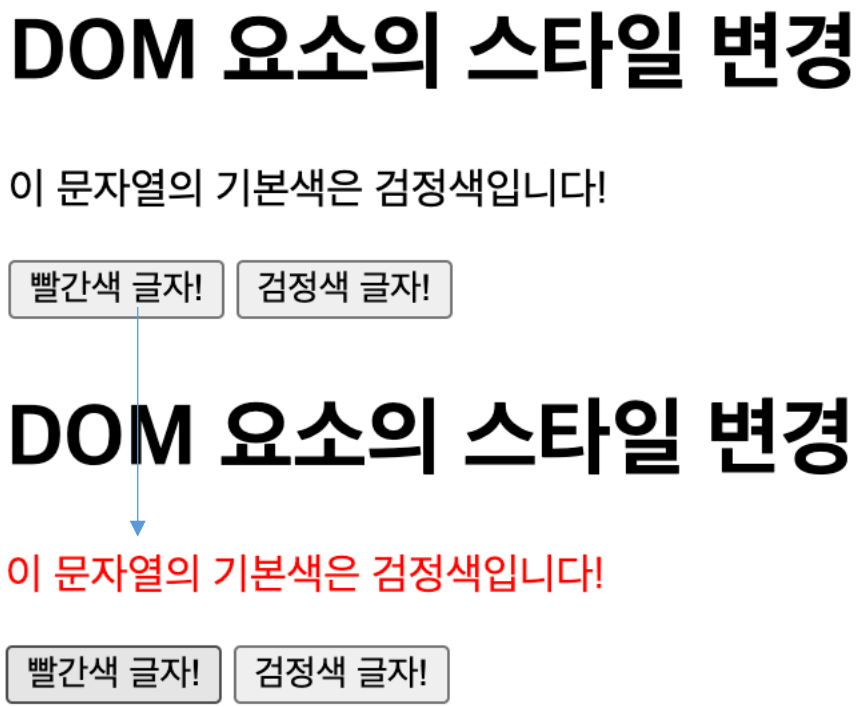
이 문장으로 바뀌었습니다!

DOM 요소의 내용변경 2

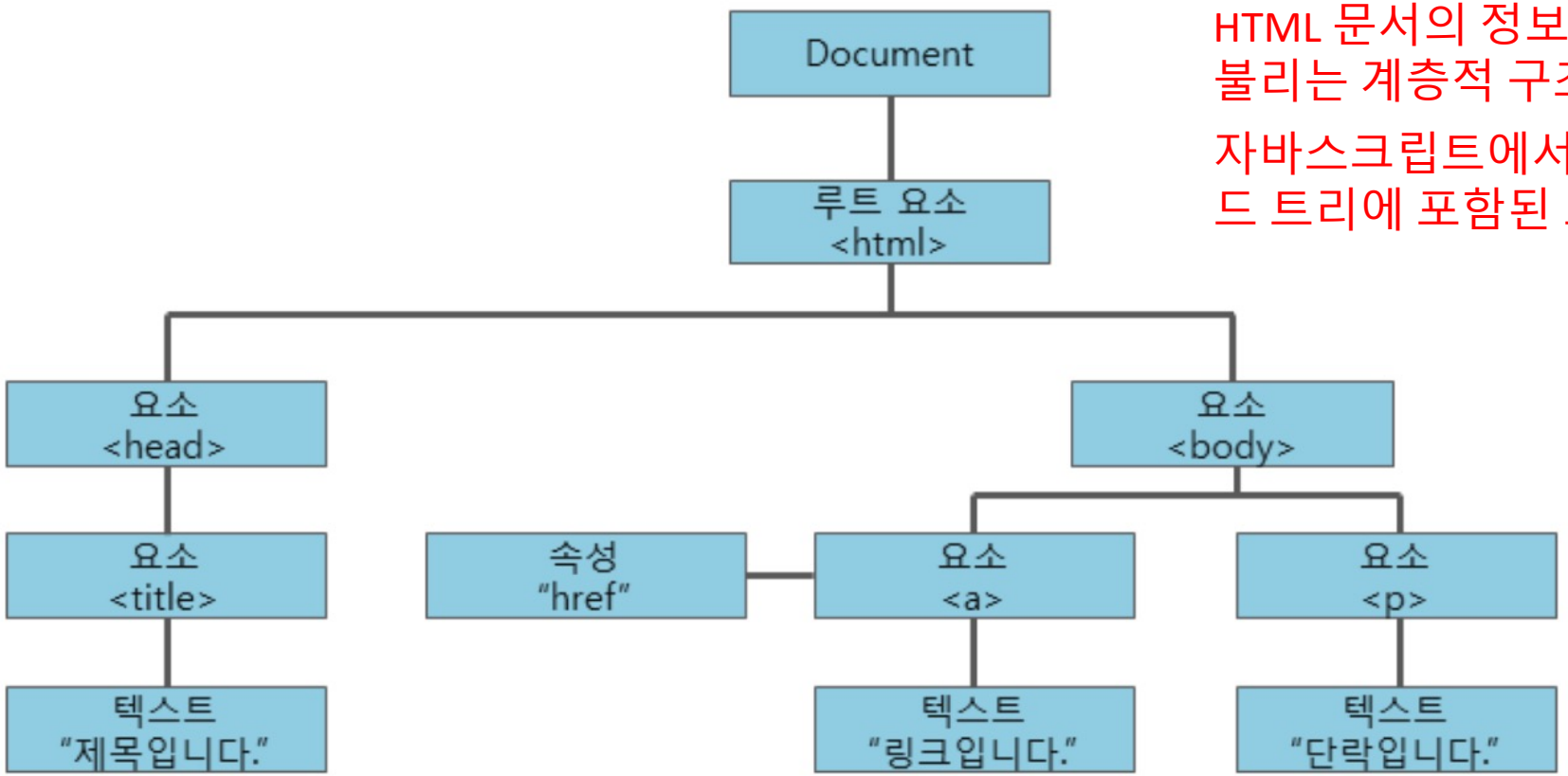
```
<!DOCTYPE html>
<html lang="ko">
<head>
	<meta charset="UTF-8">
	<title>JavaScript DOM Element</title>
</head>

<body>

	<h1>속성 이름을 이용한 속성값 변경</h1>
	<a id="link" href="http://tcpschool.com/html/intro">HTML 수업 바로 가기!</a><br><b
r>
	<button onclick="changeLink()">자바스크립트 수업으로 바꾸기!</button>

	<script>
		function changeLink() {
			var link = document.getElementById("link");
		// 아이디가 "link"인 요소를 선택함.
			link.href = "http://tcpschool.com/javascript/intro";// 해당 요소의 href
속성값을 변경함.
			link.innerHTML = "자바스크립트 수업 바로 가기!";
	// 해당 요소의 내용을 변경함.
		}
	</script>

</body>
</html>
```

DOM 요소의 스타일 변경

```
<body>

        <h1>DOM 요소의 스타일 변경</h1>
        <p id="text">이 문자열의 기본색은 검정색입니다!</p>
        <button onclick="changeRedColor()">빨간색 글자!</button
>
        <button onclick="changeBlackColor()">검정색 글자!</butto
n>

        <script>
                var str = document.getElementById("text");  // 아이
디가 "text"인 요소를 선택함.
                function changeRedColor() {
                        str.style.color = "red";
                // 해당 요소의 글자색을 빨간색으로 변경함.
                }
                function changeBlackColor() {
                        str.style.color = "black";
                // 해당 요소의 글자색을 검정색으로 변경함.
                }
        </script>

</body>
```



노드트리



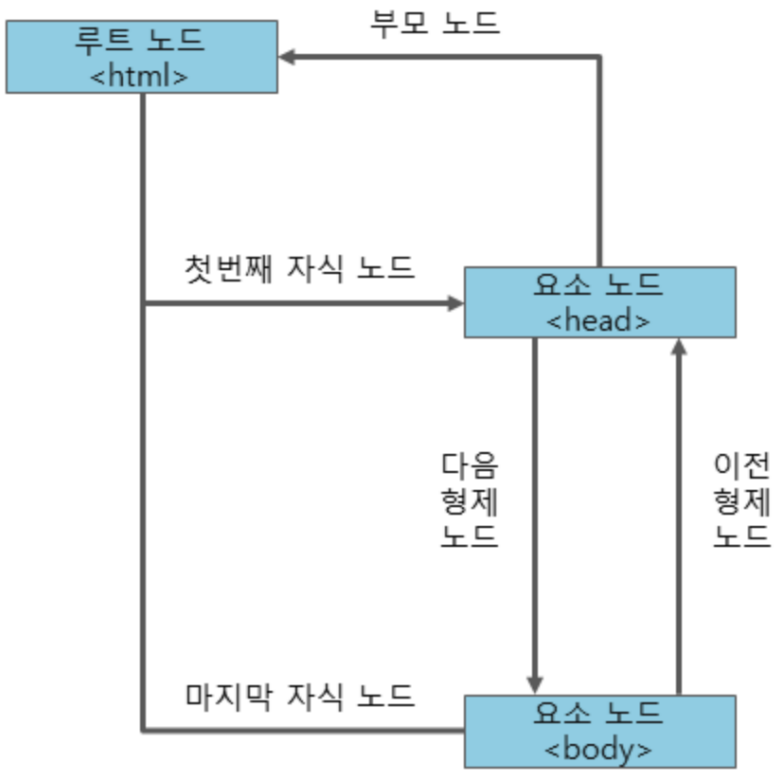
HTML 문서의 정보는 노드 트리(node tree)라고 불리는 계층적 구조에 저장

자바스크립트에서는 HTML DOM을 이용하여 노드 트리에 포함된 모든 노드에 접근할 수 있음

노드의 종류

노드	설명
문서 노드(document node)	HTML 문서 전체를 나타내는 노드임.
요소 노드(element node)	모든 HTML 요소는 요소 노드이며, 속성 노드를 가질 수 있는 유일한 노드임.
속성 노드(attribute node)	모든 HTML 요소의 속성은 속성 노드이며, 요소 노드에 관한 정보를 가지고 있음. 하지만 해당 요소 노드의 자식 노드(child node)에는 포함되지 않음.
텍스트 노드(text node)	HTML 문서의 모든 텍스트는 텍스트 노드임.
주석 노드(comment node)	HTML 문서의 모든 주석은 주석 노드임.

노드간의 관계



노드 트리의 가장 상위에는 단 하나의 **루트 노드(root node)**가 존재

루트 노드를 제외한 모든 노드는 단 하나의 **부모 노드(parent node)**만을 갖을수 있음

모든 요소 노드는 **자식 노드(child node)**를 갖을 수 있음

형제 노드(sibling node)란 같은 부모 노드를 가지는 모든 노드

조상 노드(ancestor node)란 부모 노드를 포함해 계층적으로 현재 노드보다 상위에 존재하는 모든 노드

자손 노드(descendant node)란 자식 노드를 포함해 계층적으로 현재 노드보다 하위에 존재하는 모든 노드

노드로의 접근

1. getElementsByTagName() 메소드를 이용하는 방법
2. 노드 간의 관계를 이용하여 접근하는 방법

1. 메소드를 이용

- HTML 태그 이름(tag name)을 이용한 선택
- 아이디(id)를 이용한 선택
- 클래스(class)를 이용한 선택
- name 속성을 이용한 선택

2. 노드간의 관계를 이용한 접근

1. parentNode : 부모 노드
2. childNodes : 자식 노드 리스트
3. firstChild : 첫 번째 자식 노드
4. lastChild : 마지막 자식 노드
5. nextSibling : 다음 형제 노드
6. previousSibling : 이전 형제 노드

노드에 대한 정보 가져오기

1. nodeName
2. nodeValue
3. nodeType

1. nodeName

노드	프로퍼티 값
문서 노드(document node)	#document
요소 노드(element node)	태그 이름 (영문자로 대문자)
속성 노드(attribute node)	속성 이름
텍스트 노드(text node)	#text

nodeName 예제

```
<!DOCTYPE html>
<html lang="ko">

<head>
	<meta charset="UTF-8">
	<title>JavaScript Node Access</title>
</head>
<body>
	<h1>nodeName 프로퍼티</h1>
	<p id="document"></p>
	<p id="html"></p>

	<script>
		// HTML 문서의 모든 자식 노드 중에서 두 번째 노드의 이름을 선택함.
		document.getElementById("document").innerHTML = document.childNodes[1].nodeName;
		// HTML

		// html 노드의 모든 자식 노드 중에서 첫 번째 노드의 이름을 선택함.
		document.getElementById("html").innerHTML = document.childNodes[1].childNodes[0].nodeName;
	// HEAD
	</script>

</body></html>
```

nodeName 프로퍼티

HTML

HEAD

EWHA, THE FUTURE WE CREATE

2. nodeValue

노드	프로퍼티 값
요소 노드(element node)	undefined
속성 노드(attribute node)	해당 속성의 속성값
텍스트 노드(text node)	해당 텍스트 문자열

nodeValue 예제

```
<body>

        <h1 id="heading">nodeValue 프로퍼티</h1>
        <p id="text1">텍스트</p>
        <p id="text2">텍스트</p>

        <script>
                // 아이디가 "heading"인 요소의 첫 번째 자식 노드의 노드값을 선택함.
                var headingText = document.getElementById("heading").firstChild.nodeValue;

                document.getElementById("text1").innerHTML = headingText;
                document.getElementById("text2").firstChild.nodeValue = headingText;
        </script>

</body>
```

nodeValue 프로퍼티

nodeValue 프로퍼티

nodeValue 프로퍼티

EWHA, THE FUTURE WE CREATE

3. nodeType

노드	프로퍼티 값
요소 노드(element node)	1
속성 노드(attribute node)	2
텍스트 노드(text node)	3
주석 노드(comment node)	8
문서 노드(document node)	9

nodeType 예제

```
<body>

        <h1 id="heading">nodeType 프로퍼티</h1>
        <p id="head"></p>
        <p id="document"></p>

        <script>
                // 아이디가 "heading"인 요소의 첫 번째 자식 노드의 타입을 선택함.
                var headingType = document.getElementById("heading").firstChild.nodeType;

                document.getElementById("head").innerHTML = headingType; // 3
                document.getElementById("document").innerHTML = document.nodeType; // 9
        </script>

</body>
```

nodeType 프로퍼티

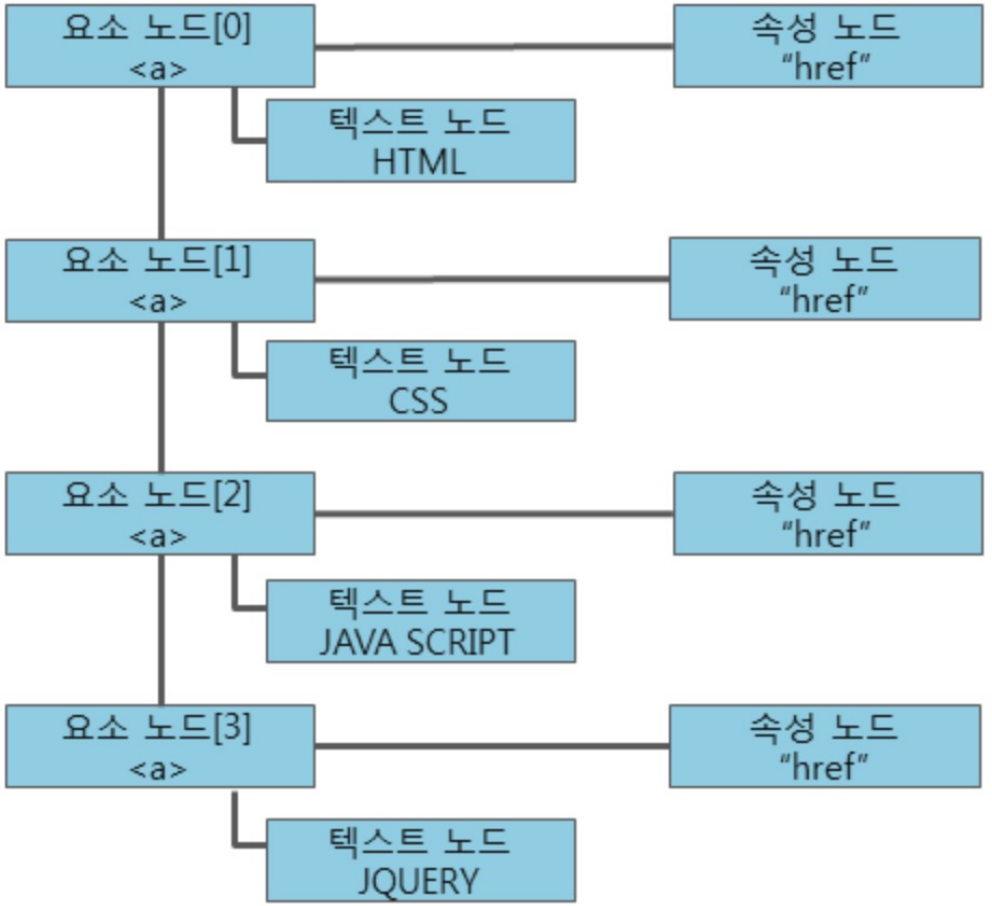
3

9

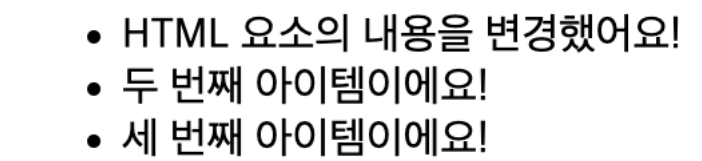
EWHA, THE FUTURE WE CREATE

노드리스트

- 노드 리스트는 getElementsByTagName() 메소드나 childNodes 프로퍼티의 값으로 반환되는 객체
- 모든 노드를 **리스트** 형태로 저장
- 리스트의 각 노드는 0부터 시작하는 인덱스를 이용하여 접근



노드리스트 예제




```
<body>

        <h1>노드 리스트</h1>
        <ul id="list"> ul 요소 다음의 텍스트
                <li>첫 번째 아이템이에요!</li>
                <li>두 번째 아이템이에요!</li>
                <li>세 번째 아이템이에요!</li>
        </ul>

        <script>
                // 아이디가 "list"인 요소의 모든 자식 노드들을 선택함.
                var listItems = document.getElementById("list").childNodes;
                // 자식 노드들 중 첫 번째 li 요소의 내용을 변경함.
                listItems[1].firstChild.nodeValue = "HTML 요소의 내용을 변경했어요!";
        </script>

</body>
```

위의 예제에서 자식 노드 중 첫 번째 요소를 선택할 때 인덱스로 0이 아닌 1을 사용
그 이유는 HTML DOM에서 각 요소 노드 다음에는 별도의 텍스트 노드가 존재

노드리스트 예제 (텍스트 노드)

```
<body>
    <h1>노드 리스트의 텍스트 노드</h1>
    <ul id="list">ul 요소 다음의 텍스트
        <li>첫 번째 아이템이에요!</li>첫 번째 li 요소 다음의 텍스트
        <li>두 번째 아이템이에요!</li>두 번째 li 요소 다음의 텍스트
        <li>세 번째 아이템이에요!</li>세 번째 li 요소 다음의 텍스트
    </ul>

    <script>
        // 아이디가 "list"인 요소의 모든 자식 노드들을 선택함.
        var listItems = document.getElementById("list").childNodes;
        // 자식 노드들 중 첫 번째 노드의 값을 출력함.
        document.write(listItems[0].nodeValue + "<br>");

        // 자식 노드들 중 두 번째 노드의 자식 노드 중 첫 번째 노드의 값을 출력함.
        document.write(listItems[1].firstChild.nodeValue + "<br>");
        // 자식 노드들 중 세 번째 노드의 값을 출력함.
        document.write(listItems[2].nodeValue);
    </script>
</body>
```

노드 리스트의 텍스트 노드

ul 요소 다음의 텍스트
- 첫 번째 아이템이에요!
  첫 번째 li 요소 다음의 텍스트
- 두 번째 아이템이에요!
  두 번째 li 요소 다음의 텍스트
- 세 번째 아이템이에요!
  세 번째 li 요소 다음의 텍스트

ul 요소 다음의 텍스트
첫 번째 아이템이에요!
첫 번째 li 요소 다음의 텍스트

각 요소 노드 다음에 또 다른 텍스트 노드가 존재

EWHA, THE FUTURE WE CREATE

노드 리스트의 length 속성

```
<body>

        <h1>노드 리스트의 length 속성</h1>
        <ul id="list">
                <li>첫 번째 아이템이에요!</li>
                <li>두 번째 아이템이에요!</li>
                <li>세 번째 아이템이에요!</li>
        </ul>
        <p id="text"></p>
        <button onclick="changeTextColor()">글자색 변경!</button>

        <script>
                var listItems = document.getElementsByTagName("li");      // 모든 <li> 요소들을 선택함.
                document.getElementById("text").innerHTML =
                        "이 노드 리스트의 길이는 " + listItems.length + "개 입니다.<br>";//모든 자식 노드들의 개수를 반환함
.

                function changeTextColor() {
                        for (var i = 0; i < listItems.length; i++) {
                        listItems[i].style.color = "orange";          // 모든 자식 노드들의 글자색을 변경함.
                        }
                }
        </script>
</body>
```

- 첫 번째 아이템이에요!
- 두 번째 아이템이에요!
- 세 번째 아이템이에요!

이 노드 리스트의 길이는 3개 입니다.

글자색 변경!

length 프로퍼티 값은 언제나 노드 리스트가 저장하고 있는 노드들의 총 개수

EWHA, THE FUTURE WE CREATE

노드의 추가

1. appendChild()
2. insertBefore()
3. insertData()

1. appendChild()

```
<body>

        <h1>appendChild() 메소드</h1>
        <h2 id="item">JavaScript</h2>
        <div id="list">
                <p>HTML</p>
                <p>CSS</p>
                <p>JQuery</p>
        </div>
        <button onclick="appendNode()">노드 추가!</button>

        <script>
                function appendNode() {
                        var parent = document.getElementById("list");          // 아이디가 "list"인 요소를 선택함.
                        var newItem = document.getElementById("item");     // 아이디가 "item"인 요소를 선택함.
                        parent.appendChild(newItem); // 해당 요소의 맨 마지막 자식 노드로 추가함.
                }
        </script>

</body>
```

2. insertBefore()

- 새로운 노드를 특정 자식 노드 **바로 앞에** 추가
- 부모노드.insertBefore(새로운자식노드, 기준자식노드);
  1. 새로운 자식 노드 : 자식 노드 리스트(child node list)에 새롭게 추가할 자식 노드를 전달
  2. 기준 자식 노드 : 새로운 노드를 삽입할 때 기준이 되는 노드로, 이 노드 바로 앞에 새로운 노드가 추가

insertBefore() 예제

```
<body>

        <h1>insertBefore() 메소드</h1>
        <h2 id="item">JavaScript</h2>
        <div id="list">
                <p>HTML</p>
                <p id="criteria">CSS</p>
                <p>JQuery</p>
        </div>
        <button onclick="appendNode()">노드 추가!</button>

        <script>
                function appendNode() {
                        var parent = document.getElementById("list");     // 아이디가 "list"인 요소를 선택함.
                        var criteriaItem = document.getElementById("criteria"); // 아이디가 "criteria"인 요소를 선택함.
                        var newItem = document.getElementById("item");            // 아이디가 "item"인 요소를 선택함.
                        parent.insertBefore(newItem, criteriaItem);          // 해당 노드를 기준이 되는 자식 노드의 바로
앞에 추가함.
                }
        </script>

</body>
```

3. insertData()

- 텍스트 노드의 텍스트 데이터에 새로운 텍스트를 추가
- 텍스트노드.insertData(오프셋, 새로운데이터);

  1. 오프셋(offset) : 오프셋 값은 0부터 시작하며, 기존 텍스트 데이터의 **몇 번째 위치부터 추가할지를** 전달

  2. 새로운 데이터 : 새로이 삽입할 텍스트 데이터를 전달

insertData() 예제

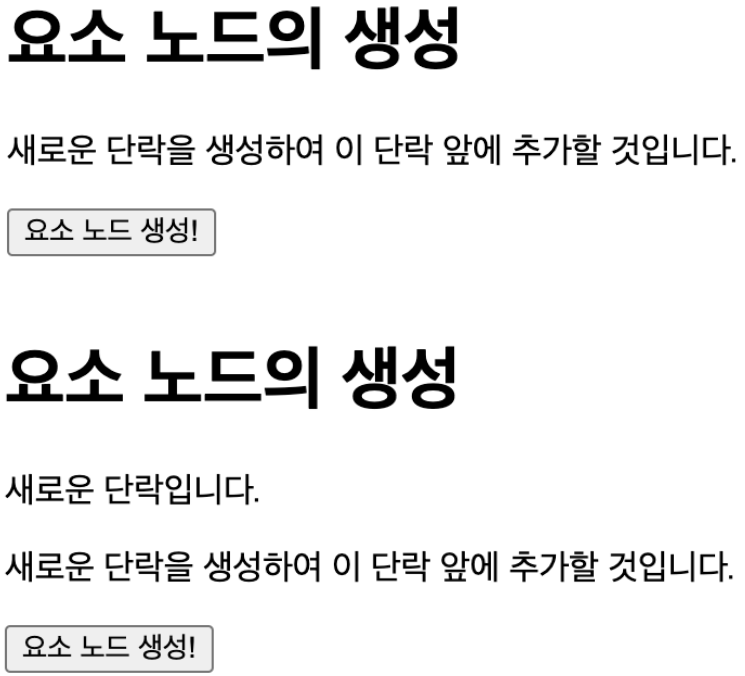
```
<body>

        <h1>insertData() 메소드</h1>
        <p id="text">지금 시간은 오후 3시입니다.</p>
        <button onclick="appendText()">텍스트 추가!</button>

        <script>
                var text = document.getElementById("text").firstChild;
        // 아이디가 "text"인 요소의 텍스트 노드를 선택함.
                function appendText() {
                        text.insertData(6, " 나른한 ");
// 텍스트 노드의 6번째 문자부터 " 나른한 "이란 텍스트를 추가함.
                }
        </script>

</body>
```

노드의 생성

1. createElement()
2. createAttribute()
3. createTextNode()

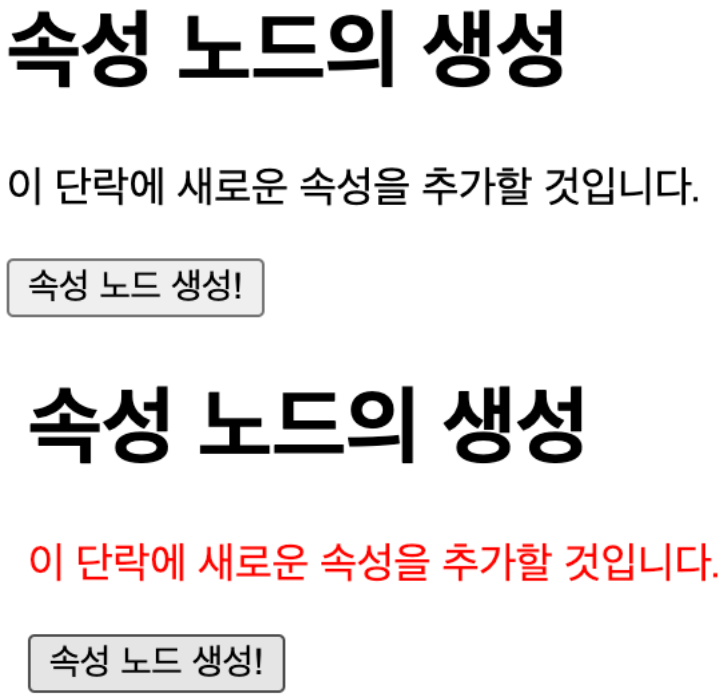
1. 요소노드의 생성: createElement()

```
<body>

        <h1>요소 노드의 생성</h1>
        <p id="text">새로운 단락을 생성하여 이 단락 앞에 추가할 것입니다.</p
        <button onclick="createNode()">요소 노드 생성!</button>

        <script>
                function createNode() {
                        var criteriaNode = document.getElementById("text");
        // 기준이 되는 요소로 아이디가 "text"인 요소를 선택함.
                        var newNode = document.createElement("p");
                // 새로운 <p> 요소를 생성함.
                        newNode.innerHTML = "새로운 단락입니다.";
                        document.body.insertBefore(newNode, criteriaNode);
        // 새로운 요소를 기준이 되는 요소 바로 앞에 추가함.
                }
        </script>

</body>
```



2. 속성노드의 생성: createAttribute()

```
<h1>속성 노드의 생성</h1>
        <p id="text">이 단락에 새로운 속성을 추가할 것입니다.</p>
        <button onclick="createNode()">속성 노드 생성!</button>

        <script>
                function createNode() {
                        var text = document.getElementById("text");
                // 아이디가 "text"인 요소를 선택함.
                        var newAttribute = document.createAttribute("style");
        // 새로운 style 속성 노드를 생성함.
                        newAttribute.value = "color:red";
                        text.setAttributeNode(newAttribute);
                        // 해당 요소의 속성 노드로 추가함.
                }
        </script>
```



이미 존재하는 요소 노드에 속성 노드를 생성하고자 할 때에는 setAttribute() 메소드를 사용

3. 텍스트노드의 생성

```
<h1>텍스트 노드의 생성</h1>
        <button onclick="createNode()">텍스트 노드 생성!</button>
        <p id="text"></p>

        <script>
                function createNode() {
                        var elementNode = document.getElementById("text");
                        // 아이디가 "text"인 요소를 선택함.
                        var newText = document.createTextNode("새로운 텍스트에요! ");
        // 새로운 텍스트 노드를 생성함.
                        elementNode.appendChild(newText);
                                        // 해당 요소의 자식 노드로 추가함.
                }
        </script>
```

텍스트 노드의 생성

텍스트 노드 생성!

텍스트 노드의 생성

텍스트 노드 생성!

새로운 텍스트에요!

노드의 제거

1. removeChild()
2. removeAttribute()

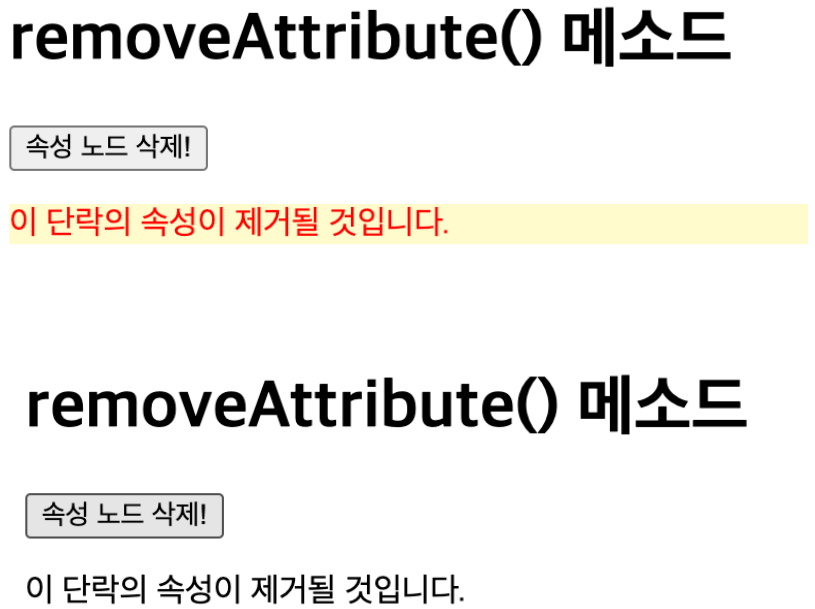
1. removeChild()

- 자식 노드 리스트에서 특정 자식 노드를 제거

```
<h1>removeChild() 메소드</h1>
        <button onclick="remove()">요소 노드 삭제!</button>
        <div id="list">
                <p>HTML</p>
                <p id="item">CSS</p>
                <p>JavaScript</p>
        </div>

        <script>
                function remove() {
                        var parent = document.getElementById("list");
// 아이디가 "list"인 요소를 선택함.
                        var removedItem = document.getElementById("item");
// 아이디가 "item"인 요소를 선택함.
                        parent.removeChild(removedItem);
                // 지정된 요소를 삭제함.
                }
        </script>
```

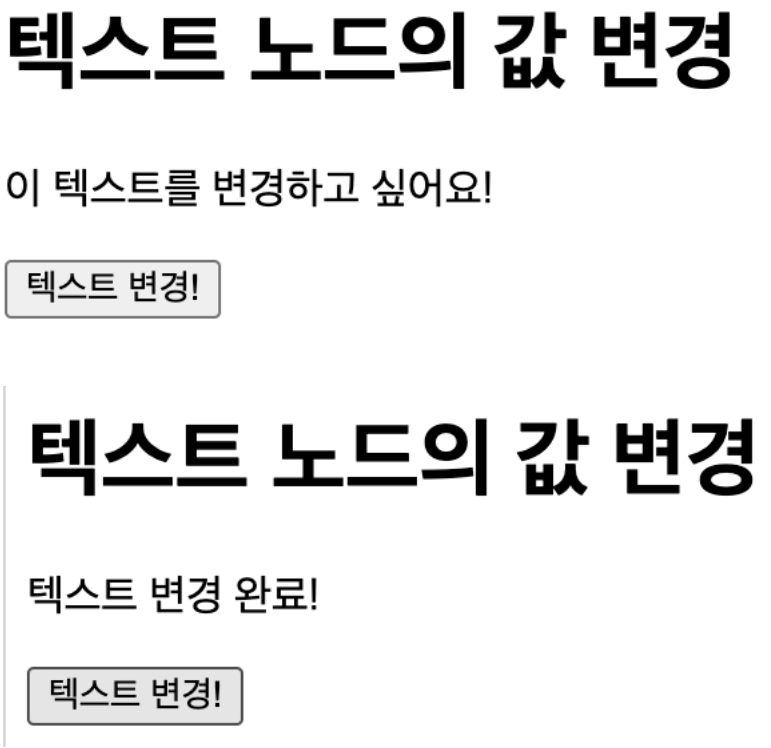
2. removeAttribute()

```
<h1>removeAttribute() 메소드</h1>
        <button onclick="remove()">속성 노드 삭제!</button>
        <p id="text" style="color:red; background-color:lemonchiffon;">
이 단락의 속성이 제거될 것입니다.</p>

        <script>
                function remove() {
                        var text = document.getElementById("text");
        // 아이디가 "text"인 요소를 선택함.
                        text.removeAttribute("style");
                        // 해당 요소의 "style" 속성을 제거함.
                }
        </script>
```



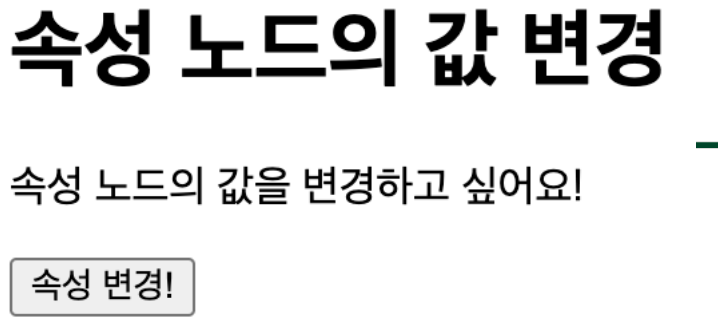
텍스트노드의 값 변경

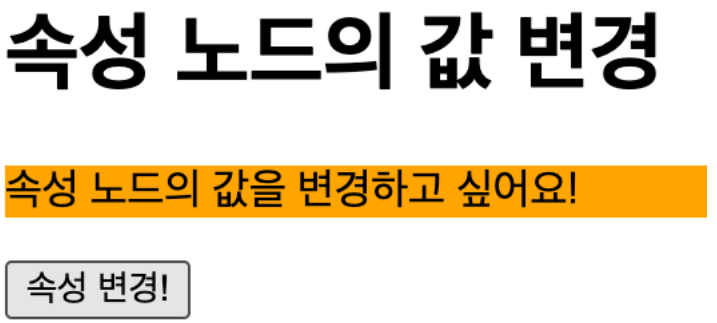
```
<h1>텍스트 노드의 값 변경</h1>
        <p id="text">이 텍스트를 변경하고 싶어요!</p>
        <button onclick="changeText()">텍스트 변경!</button>

        <script>
                var para = document.getElementById("text");
        // 아이디가 "text"인 요소를 선택함.
                function changeText() {
                        para.firstChild.nodeValue = "텍스트 변경 완료!";
                }
        </script>
```



속성노드의 값 변경

```
<style>
        .para {
        background-color : orange;
        }
</style>

<h1>속성 노드의 값 변경</h1>
        <p>속성 노드의 값을 변경하고 싶어요!</p>
        <button onclick="changeAttribute()">속성 변경!</button>

        <script>
                var para;
                function changeAttribute() {
                        // 모든 <p> 요소중에서 첫 번째 요소에 클래스 속성값으로 "para"를 설정함.
                        document.getElementsByTagName("p")[0].setAttribute("class", "para");
                        // 클래스가 설정되면 해당 클래스에 설정되어 있던 스타일이 자동으로 적용됨.
                }
        </script>
```





요소노드의 교체

- 기존의 요소 노드를 새로운 요소 노드로 교체
- 교체할노드 = 부모노드.replaceChild(새로운자식노드, 기존자식노드);
  1. 새로운 자식 노드 : 자식 노드 리스트에 새롭게 추가할 요소 노드를 전달
  2. 기존 자식 노드 : 자식 노드 리스트에서 제거할 요소 노드를 전달

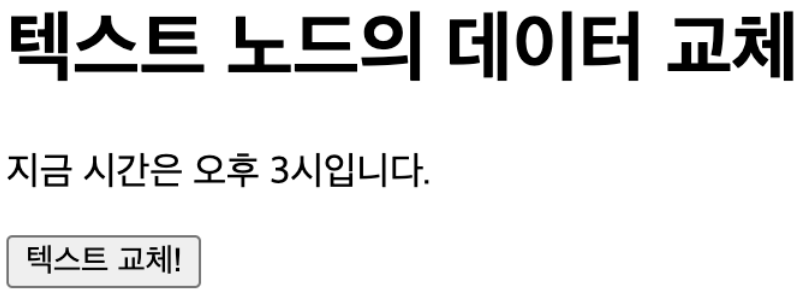
요소노드의 교체 예제

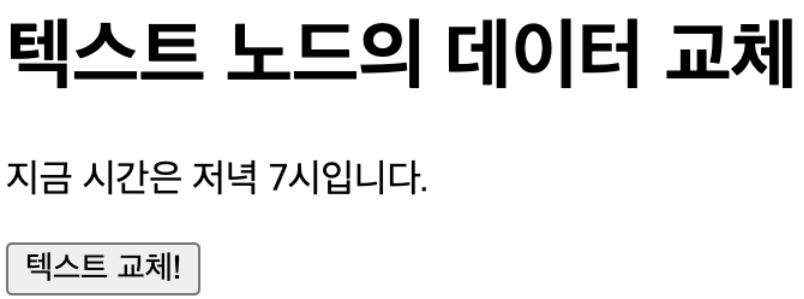
```
<h1>요소 노드의 교체</h1>
        <div id="parent">
                <p id="first">첫 번째 요소입니다.</p>
                <p>두 번째 요소입니다.</p>
        </div>
        <p id="third">세 번째 요소입니다.</p>
        <button onclick="changeNode()">요소 노드 교체!</button>

        <script>
                var parent = document.getElementById("parent");     //
부모 노드를 선택함.
                var first = document.getElementById("first");
                var third = document.getElementById("third");
                function changeNode() {
                        parent.replaceChild(third, first);
        // first 요소를 삭제하고, 그 대신 third 요소를 삽입함.
                }
        </script>
```

텍스트노드의 데이터 교체

- 텍스트노드.replaceData(오프셋, 교체할문자수, 새로운데이터);
  1. 오프셋(offset) : 오프셋 값은 0부터 시작하며, 기존 텍스트 데이터의 몇 번째 문자부터 교체할지를 전달
  2. 교체할 문자 수 : 기존 텍스트 노드의 데이터로부터 교체할 총 문자 수를 전달
  3. 새로운 데이터 : 새로이 삽입할 텍스트 데이터를 전달

텍스트노드의 데이터 교체 예제






```
<h1>텍스트 노드의 데이터 교체</h1>
        <p id="text">지금 시간은 오후 3시입니다.</p>
        <button onclick="changeText()">텍스트 교체!</button>

        <script>
                var text = document.getElementById("text").firstChild;
                // 아이디가 "text"인 요소의 텍스트 노드를 선택함.
                function changeText() {
                        text.replaceData(7, 4, "저녁 7");
                        // 텍스트 노드의 7번째 문자부터 4개의 문자를 "저녁 7"로 교체함.
                }
        </script>
```

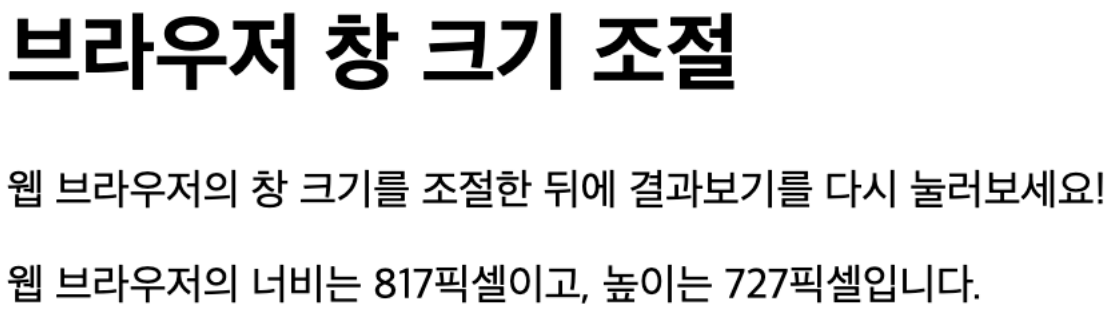
JavaScript 문법 3: 브라우저객체

-- 튜토리얼 by w3schools & tcpschool.com

Window

- 웹 브라우저의 창(window)을 나타내는 객체
- 문서 객체 모델(DOM)의 요소들도 모두 window 객체의 프로퍼티가 됨

Window: 브라우저 창크기




```
<h1>브라우저 창 크기 조절</h1>
	<p>웹 브라우저의 창 크기를 조절한 뒤에 결과보기를 다시 눌러보세요!</p>

	<script>
		var windowWidth = window.innerWidth;
		//document.documentElement.clientWidth or document.body.clientWidth

		var windowHeight = window.innerHeight;
		// document.documentElement.clientHeight or document.body.clientHeight

		document.write("웹 브라우저의 너비는 " + windowWidth + "픽셀이고, 높이는 "
+ windowHeight + "픽셀입니다.");
	</script>
```

Window: 브라우저 창열기

```
<h1>브라우저 새 창 열기</h1>
        <button onclick="openWindow()">새로운 창 열기</button>

        <script>
                var newWindowObj;
                // 변수 strWindowFeatures를 통해 새롭게 여는 웹 브라우저 창의 옵션들을 일일이 설정할 수 있음.
                var strWindowFeatures = "menubar = yes, location = yes, resizable = yes, scrollbars = yes, status
= yes";
                function openWindow() {
                        newWindowObj = window.open("https://www.google.com");
                }
        </script>
```

Window: 브라우저 창닫기

```
<h1>브라우저 창 닫기</h1>
        <button onclick="openWindow()">새로운 창 열기</button>
        <button onclick="closeNewWindow()">새로 연 창 닫기</button>

        <script>
                var newWindowObj;
                var strWindowFeatures = "menubar = yes,location = yes,resizable = yes,scrollbars = yes,status = yes";
                function openWindow() {
                        newWindowObj = window.open("/html/intro", "HTML 개요", strWindowFeatures);
                }
                function closeNewWindow() {       // 새롭게 연 브라우저 창을 window 객체를 이용하여 다시 닫을 수 있음.
                        newWindowObj.close();
                }
        </script>
```


EWHA, THE FUTURE WE CREATE

Location

- 현재 브라우저에 표시된 HTML 문서의 주소를 얻거나, 브라우저에 새 문서를 불러올 때 사용
- Window 객체의 location 프로퍼티와 Document 객체의 location 프로퍼티에 같이 연결
- 예제

```
document.write("현재 문서의 주소는 " + location.href + "입니다.");
document.write("현재 문서의 호스트 이름은 " + location.hostname + "입니다.");
document.write("현재 문서의 파일 경로명은 " + location.pathname + "입니다.");
```

현재 문서의 주소는 https://www.w3schools.com/html/tryit.asp?filename=tryhtml_default입니다.
현재 문서의 호스트 이름은 www.w3schools.com입니다.
현재 문서의 파일 경로명은 /html/tryit.asp입니다.

History: back(), go()

- 브라우저의 히스토리 목록에서 바로 이전 URL로 이동

```
<button onclick="goBack()">이전 페이지로 가기</button>
...
<script>
    function goBack() {
        window.history.back();
    }
</script>

<button onclick="go()">이전 페이지로 가기</button>
...
<script>
    function go() {
        window.history.go(-1);
    }
</script>
```

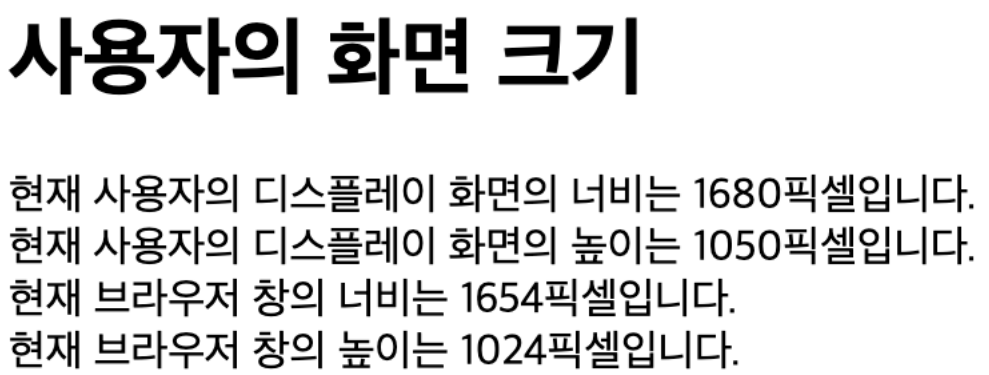
History: forward()

- 브라우저의 히스토리 목록에서 바로 다음 URL로 이동

```
<button onclick="goForward()">다음 페이지로 가기</bu
tton>
...
<script>
    function goForward() {
        window.history.forward();
    }
</script>
```

Screen

```
<h1>사용자의 화면 크기</h1>

        <script>
                document.write("현재 사용자의 디스플레이 화면의 너비는 " + screen.width + "픽셀입니다.<
br>");
                document.write("현재 사용자의 디스플레이 화면의 높이는 " + screen.height + "픽셀입니다.
<br>");
                document.write("현재 브라우저 창의 너비는 " + window.outerWidth + "픽셀입니다.<br>");
                document.write("현재 브라우저 창의 높이는 " + window.outerHeight + "픽셀입니다.<br>");
        </script>
```



Navigator

- 브라우저 공급자 및 버전 정보 등을 포함한 브라우저에 대한 다양한 정보를 저장

```
<h1>현재 브라우저의 버전</h1>

        <script>
                document.write("현재 사용 중인 브라우저의 버전 정보는 " + navigator.appVersion + "
입니다.<br><br>");

        </script>
```

현재 브라우저의 버전

현재 사용 중인 브라우저의 버전 정보는 5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safari/537.36입니다.

팀프로젝트 프로포절 발표

- 평가: 교수 40% + 조교 40% + 동료평가 20%
- 예상되는 내용
  - 프론트엔드 설계내용 (그래픽 툴 or hand-writing)
  - 사용하고자하는 Opensource software (or platform)
    - Frontend: HTML, CSS, JS
    - Backend: Flask
    - 그외에 사용하고 하는 OSS 있다면 추가
  - 업무분담
  - 타임라인
- 발표순서: 랜덤하게 선택하여 사이버캠퍼스에 공지예정
- 평가기준 (40점 만점)
  - 발표점수 (10점)
  - 프론트엔드 디자인이 사용자친화적인지 (10점)
  - 프론트엔드 디자인이 프로젝트 요구사항을 모두 반영하는지 (10점)
  - 업무분담 및 타임라인 실현가능성 (10점)

Coming up

- 이번주 과제 없고 수업시간에 배운 예제들 실습해보고 프로포절 발표 준비 팀회의 진행하기
- Summary of TO-DO
  - 팀프로젝트 화면설계 (Part1) 사이버캠퍼스 제출 (10/18)
  - 팀프로젝트 화면설계 (Part2) 사이버캠퍼스 제출 (11/9)
- 10/19 프로포절 발표