7장 내장객체 사용하기

내장 객체 (Built-in Object)란 브라우저의 자바스크립트 엔진에 내장된 객체를 말합니다. 내장 객체의 종류에는 많은 객체들이 존재하는데, 그중 가장 많이 사용되는 객체를 알아보도록 하겠습니다.

우리가 많이 사용하는 객체는 앞에서 살펴본 배열(Array)객체 이외에 날짜(Date)객체, 수학(Math)객체 등이 있습니다. 각각의 객체에는 속성과 메서드가 존재합니다.

7-1 Date(날짜)객체 사용하기

Date객체는 자바스크립트에서 날짜와 시간을 관리하는 객체입니다. 우리는 6장에서 객체생성에 대해 알아보았습니다. 내장객체는 따로 선언하지 않고 사용할 수 있는 객체도 있지만, 지금 우리가 알아볼 Date객체는 기본 객체원본에 자신의 인스턴스 객체를 생성하여 사용합니다.

객체 생성하기

```
var 인스턴스 이름 = new Date();
```

실습해 봅시다.()

```
10. ❶ var Today =  new Date(); ❷
11.      ❸ console.log(Today);
12.      ❹ console.log(Today.getDay());
```

❶ 사용자가 사용할 Date객체 이름을 선언합니다.

❷ new 연산자를 이용하여 새로운 Date객체를 생성합니다. 이때 인수의 값이 존재하지 않으며 현재의 날짜와 시간을 기준으로 Date객체가 생성됩니다.

❸ console창에 만들어진 객체 Today를 출력합니다.

❹ console창에 만들어진 Date객체 메서드를 이용하여 객체 Today의 요일 값을 출력해 봅니다.

만약에 특별한 날짜를 기준으로 생성하기는 원한다면 인수에 원하는 날짜를 입력하면 된다.

```
var 인스턴스 이름 = new Date( '년도/월/일' );
var 인스턴스 이름 = new Date( 년도,월-1,일 );
```

get메서드 그룹

날짜객체의 메서드는 두 가지 그룹으로 나누어 사용합니다. get-메서드() set-메서드() 그룹으로 나눌 수 있습니다. get-메서드() 그룹은 대부분 서버의 날짜 정보를 가져와 사용되고, set-메서드()그룹은 사용자의 필요 값이 설정할 때 사용합니다. 두 그룹의 차이를 보면 get-메서드()그룹은 서버의 정보가 기준이기 때문에 값의 설정이 컴퓨터 표기 위주로 설정되어 있으며 set-메서드()그룹은 사용자의 설정값이 그대로 적용됩니다. 예를 들면 get-메서드()그룹에서 요일을 담당하는 명령어는 .getDay()입니다. 앞의 예제에서 본 바와 같이 각국에서 사용되는 요일의 명칭이 다르므로 컴퓨터 설정을 기준으로 0~6까지 Number형 값으로 설정되어 있습니다. 하지만 set-메서드()그룹에서는 요일을 설정된 날짜에 따라 자동 셋팅 되기 때문에 따로 요일을 정하는 메서드가 없습니다. 그럼 get-메서드()그룹을 먼저 정리해 봅시다.

메소드	설명
getDate( )	날짜 정보에서 일(date)의 정보를 가져옵니다. 1부터 31일까지 해당 달의 일를 가져옵니다.
getDay( )	날짜 정보에서 요일(0-6)을 숫자로 가져옵니다. 0:일요일, 1: 월요일, 2:화요일, 3:수요일, 4:목요일, 5:금요일, 6:토요일
getFullYear( )	날짜 정보 중 년도를 4자리 수로 가져옵니다.
getHours( )	시간의 정보를 가져옵니다. (0-23)
getMinutes( )	분의 정보를 가져옵니다. (0-59)
getMonth( )	날짜 정보 중 월의 정보를 숫자로(0~11) 가져옵니다. 0:1월, 1:2월, 2:3월 ……,10:11월, 11:12월
getSeconds( )	초의 정보를 가져옵니다. (0-59)
getMilliseconds( )	1초를 0~999의 숫자로 밀리초로 반환하여 가져옵니다.
getTime( )	1970년 1월 1일 이후의 0시 0분 0초를 밀리초로 표시합니다.
밀리초란 1/1000초를 로 환산하는 법입니다.	

set그룹 메서드 사용하기

set그룹은 생성된 날짜 인스턴스에 날짜 정보를 설정할 때 사용합니다. 메서드를 먼저 살펴봅시다.

메서드	설명
setDate( )	1 - 31의 숫자로 일를 지정합니다.
setFullYear()	4자리 숫자로 연도를 지정합니다.(옵션으로 달,일도 가능합니다.)
setHours()	0-23의 숫자로 시간을 지정합니다.
setMilliseconds()	밀리초를 지정한다.
setMonth()	0~11의 숫자로 월을 지정합니다.
setSeconds()	0~59의 숫자로 초를 지정한다.
setTime()	1970년 1월 1일 이후의 0시 0분 0초를 밀리초로 설정합니다.
toLocaleString()	11/09/99 12:43:22 형태로 시간과 날짜를 문자로 지정
toSource()	Date() 객체의 원형을 반환한다.
toString()	시간과 날짜를 문자열로 반환한다.

7-2 Math(수학)객체 사용하기

Math객체는 수학과 관련된 기능과 속성을 제공합니다. 물론 우리가 사용하는 연산자 중에는 연산연산자가 있지만, 일반적인 계산이 아닌 최댓값, 최젓값, 또는 random(난수) 필요한 때도 있고 반올림 반 내림이 필요할 때도 있습니다. 이러한 수학적인 부분을 도와주는 시스템이 수학객체입니다.

Math()객체는 다른 객체처럼 본 객체의 인스턴스를 만들어서 사용할 수도 있지만, 인스턴스를 만들지 않고 직접 객체에 메서드를 연결해 사용하기도 합니다.

객체생성

```
var num = Math.pow(2,3);
```

메서드

수학객체의 메서드를 정리해 봅시다. 이 객체의 경우에는 많은 수학적인 계산을 도와주는 객체가 있습니다.

메소드	설명
abs(number)	숫자의 절댓값을 반환합니다.
ceil(number)	숫자의 소수점 부분을 무조건 반올림하여 줍니다.
floor(number)	숫자의 소수점 부분을 무조건 삭제해 줍니다.
round(numbe)	숫자의 소수점 부분을 반올림 또는 반 내림해 줍니다.

무작위수(난수) 만들기

random()	0과1 사이의 난수 발생값을 반환해 주는 메소드

random()메서드는 0 ~ 0.99999의 수를 무작위로 배출하는데 우리는 0 ~ 1 사이로 생각하면 문제가 되지 않는다. 아마도 계산을 생각할 때 이렇게 계산을 하면 편합니다.

먼저 무작위 수를 만들어 봅시다.(실습 파일: 07/07_05.html. 완성 파일: 07/all/07_05.html)

```
10     //무작위:random
11    document.write(Math.random(),'<br>');
```

random() 메서드를 이용하여 난수를 만들어 보았습니다. 브라우저에서 다시 새로고침을 하면 다

른 수가 축출되어 나오게 됩니다.

그럼 이번에는 범위를 정하여 무작위 수를 축출해 봅시다. (0부터 최대수)

Math.random() x 최대 수

0부터 최대 수 500까지의 범위에서 무작위 수를 축출해 봅시다.

```
13    //0~500
14    document.write(Math.random()*500,'<br>');
```

다음에는 최소 수부터 최대의 수까지의 범위는

Math.random() x (최대 수-최소 수) + 최소수

입니다. 그럼 100부터 500까지의 무작위 수를 축출해 봅시다.

```
16    //최소 ~최대의 숫자
17    document.write(Math.random()*(500-100)+100);
```

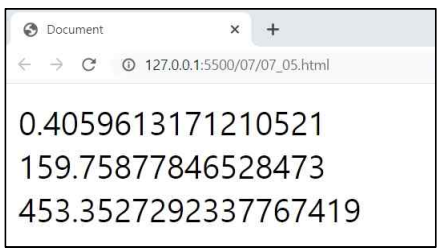
이제는 소수점을 없애기를 하면 floor()를 사용할지 아니면 ceil()을 사용할지 결정하면 됩니다.

Math.ceil(Math.random() x (최대 수-최소 수) + 최소수)

을 사용하여 봅시다.

```
21    document.write(Math.ceil(Math.random()*(500-100)+100));
```

마지막으로 출력값을 확인해 보면



그럼 마지막으로 최댓값과 최솟값을 받아 봅시다.

max(number,number)	여러 숫자 중 큰 수를 반환 해 줍니다.
min(number,number)	여러 숫자 중 작은 수를 반환 해 줍니다.

문서객체(DOM) 사용하기

DOM(문서)객체 란?

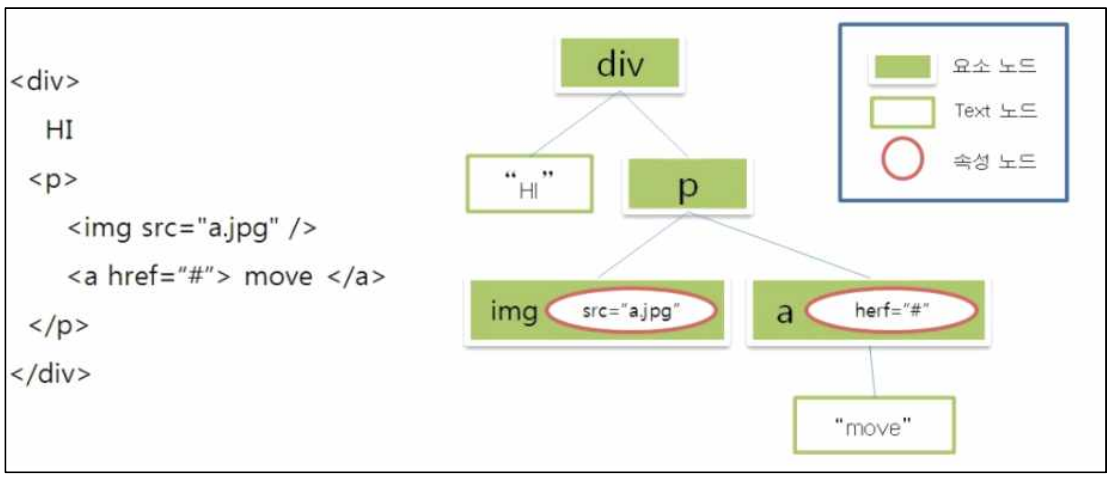
Dom객체란 Document Object Model의 약자로 우리는 DOM(돔)이라고 부릅니다. 이 말을 해석해 보면 DOM을 사용하여 웹 문서의 모든 요소를 해석할 수 있다는 뜻입니다. 이를 다시 설명하면 DOM을 통하여 문서를 해석하고 분석하여 조작할 수 있도록 하는 시스템이라고 풀이 할 수 있습니다.

노드란

HTML 문서를 구성하고 있는 개별적인 단위를 말하며 노드의 집합은 Dom이 되고, Dom을 분해하면 노드가 나옵니다.

노드의 종류

- 요소 노드(element node)
- 텍스트 노드(text node)
- 속성 노드(attribute node)입니다.



DOM트리

HTML 문서를 계층적으로 구성된 형태로 도식화 해 보면 나무에서 뿌리가 뻗어나가는 모습과 비슷하여 'DOM Tree' 또는 'Node Tree'라고 합니다.

DOM 탐색하기

DOM 트리 상단의 노드들은 document가 제공하는 프로퍼티를 사용해 접근할 수 있습니다.

<html> = document.documentElement

document를 제외하고 DOM 트리 꼭대기에 있는 문서 노드는 <html> 태그에 해당하는 document.documentElement입니다.

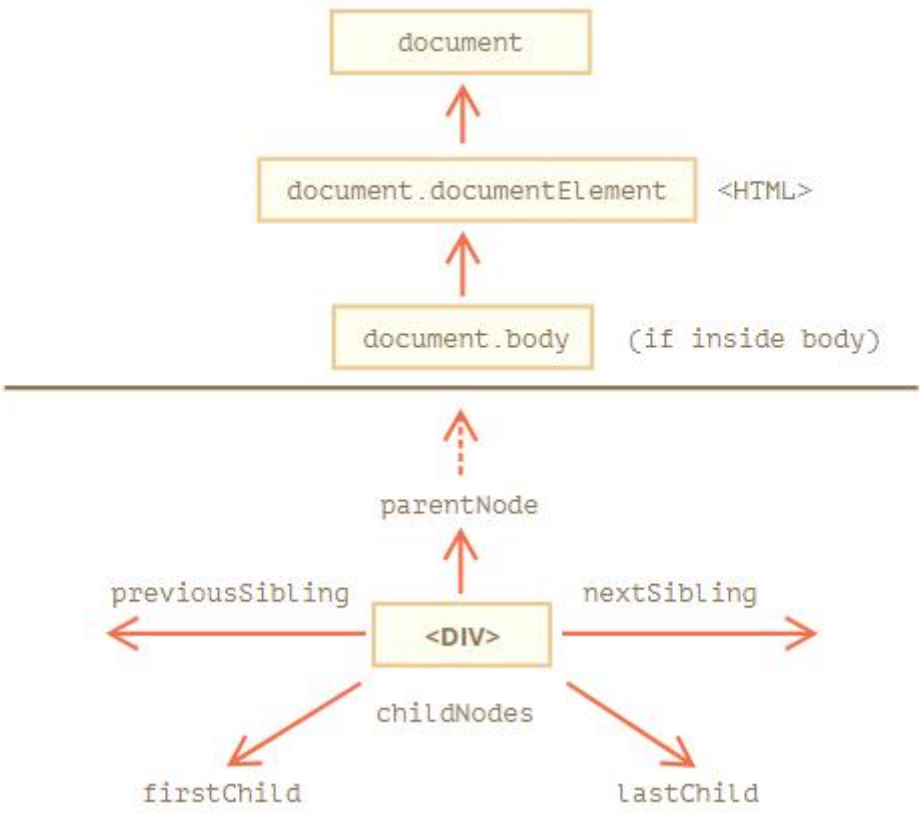
<body> = document.body

document.body는 <body> 요소에 해당하는 DOM 노드로, 자주 쓰이는 노드 중 하나입니다.

<head> = document.head

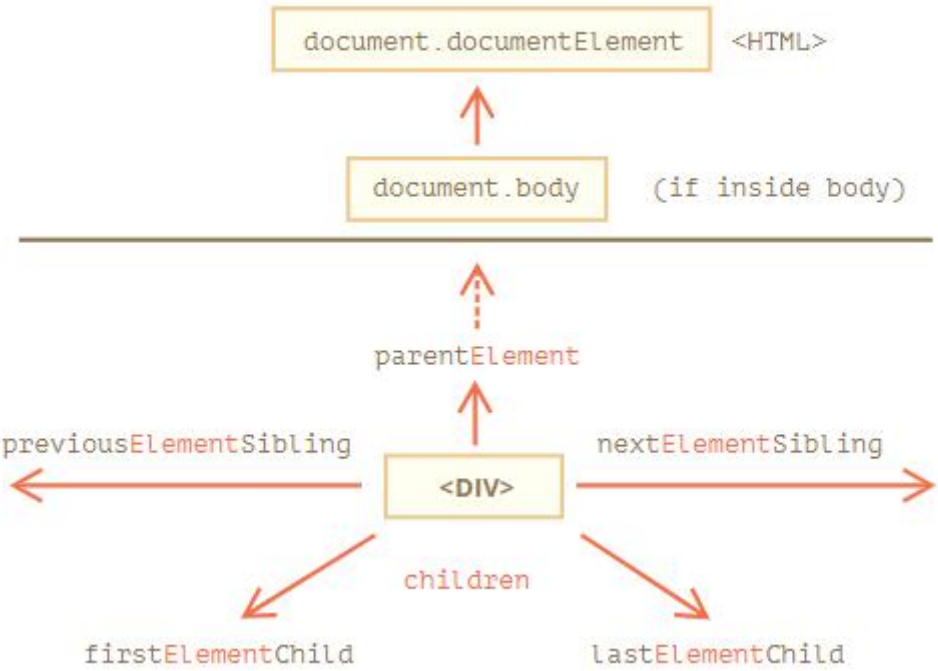
<head> 태그는 document.head로 접근할 수 있습니다.

DOM 탐색하기



document.body.parentNode
document.head.nextSibling
document.body.previousSibling

요소 간 이동



8-2 DOM 요소 선택하기

DOM요소에 접근하는 방법은 여러 가지가 있습니다. CSS로 생각하면 선택자(selector)라고 보면 됩니다. 약간의 차이가 있다면 CSS에서는 선택자로 지칭할 수 있지만 자바스크립트에서는 이를 문장을 만들기 위한 시작점이라고 보면 좋습니다. 이곳에서부터 내가 할 일을 시작하겠다는 끝이라고 보면 됩니다.

메서드	설명
getElementById('아이디명');	아이디명으로 DOM요소 선택하기
getElementsByTagName('요소명');	태그명으로 DOM요소 선택하기(여러요소가 있을 경우에는 배열로 정리하여 선택된다.)
유사배열 getElementsByClassName('클라스명')	클라스명으로 DOM요소 선택하기(여러요소가 있을 경우에는 배열로 정리하여 선택된다.)
유사배열 getElementsByTagName('태그이름')	
querySelector('css선택자')	css선택자로 DOM요소 선택하기(여러요소가 있을 경우에는 첫 번째 요소만 선택된다.)
유사배열 querySelectorAll('css선택자')	같은 css선택자 모두 선택하기(여러요소가 있을 경우에는 배열로 정리하여 선택된다.)

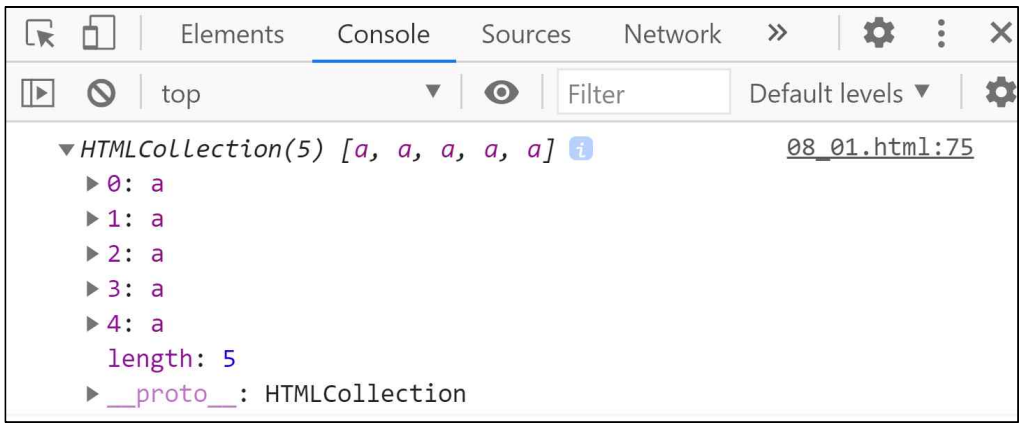
초장기의 자바스크립트는 DOM의 요소의 선택이 많이 불편한 점이 있었습니다. 그러나 자바스크립트의 버전이 업그레이드 되면서 querySelector가 만들어 지면서 우리는 DOM의 요소 쉽게 선택할 수 있습니다.

1. 아이디 값으로 선택하기

```
71    const main = document.getElementById('skill');
72    console.log(main);
```

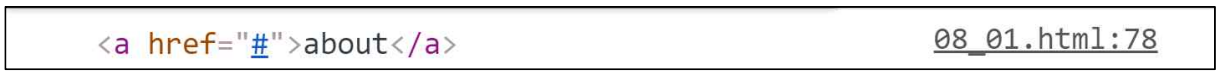
2. 요소명으로 선택하기

```
74    const sub = document.getElementsByTagName('a');
75    console.log(sub);
```



만약에 여러요소가 있을 경우에는 배열로 정리하여 선택이 됩니다. 우리를 그 배열을 이용하여 각각의 요소를 선택할 수 있습니다.

```
77    const sub2 = document.getElementsByTagName('a')[2];
78    console.log(sub2);
```



만약에 이렇게 된 요소에 스타일을 넣어 봅시다.

변수 sb2의 세 번째 a를 선택하여 style속성의 backgroundColor의 값을 red로 지정하면

```
79    sub2.style.backgroundColor= 'red';
```

로 선택이 됩니다.

3. css 선택자 사용하기

css 선택자를 사용하면 이제까지 DOM의 요소 노드로 사용하여 단조롭게 선택되던 방식에서 속성노드, text노드를 사용할 수 있게 됩니다. querySelector() 와 querySelectorAll()의 차이를 보면 querySelector()는 단 하나의 요소를 선택한다면 querySelectorAll()은 조건이 맞는 모든 요소를 선택할 수 있습니다. 그래서 querySelectorAll()도 여러요소가 선택이 되면 배열로 정리하여 선택이 됩니다. 우리를 그 배열을 이용하여 각각의 요소를 선택할 수 있습니다.

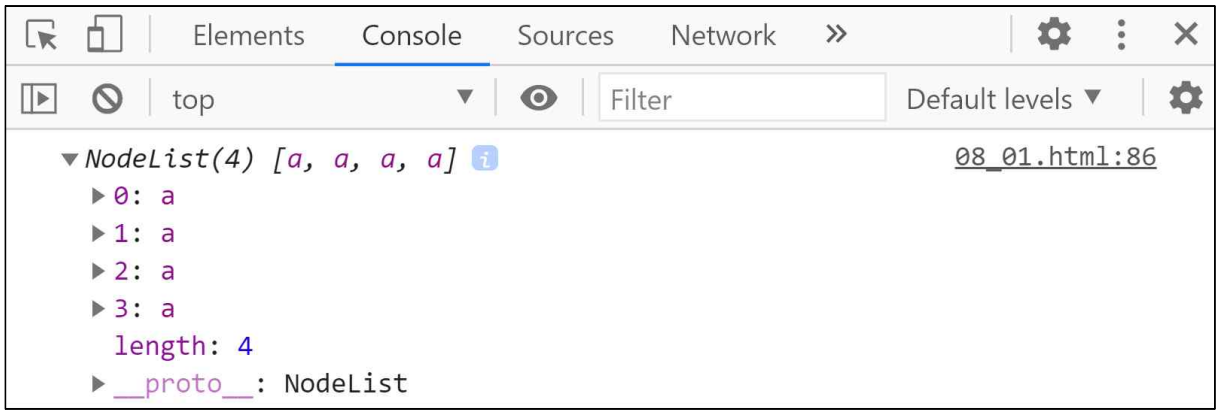
querySelector()를 이용하여 #skill>dl을 선택하여 스타일을 바꾸어 봅시다.

```
81    const se01 = document.querySelector('#skill>dl');
82    console.log(se01);
83    se01.style.backgorundColor = 'yellow';
```

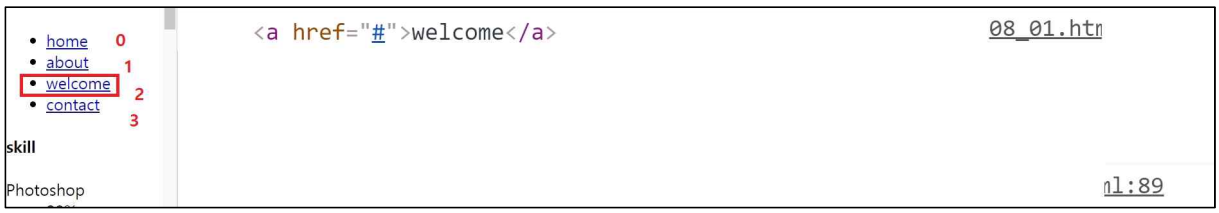
그럼 이번에는 ❶querySelectorALL()을 이용하여 nav>ul>li>a을 선택하여 출력하고 ❷변수 se03에는 nav>ul>li>a중 세 번째 a를 선택하여 text노드를 바꾸어 봅시다.

```
85    const se02 = document.querySelectorAll('nav>ul>li>a'); ❶
86    console.log(se02);
87
88    const se03 = document.querySelectorAll('nav>ul>li>a')[2];❷
89    console.log(se03);
90    se03.innerHTML = 'welcome';
```

❶의 출력을 보면 총 4개의 a요소 노드가 선택이 되었습니다.



❷의 결과를 확인해 보면 text노드가 변경 되어 있습니다.



속성	형식	설명
innerHTML	선택자.innerHTML	선택된 요소안의 노드를 가져옵니다.
	선택자.innerHTML = 값	선택된 요소안의 노드값을 변경합니다.
outerHTML	선택자.outerHTML	선택된 요소의 밖의 노드를 가져옵니다.

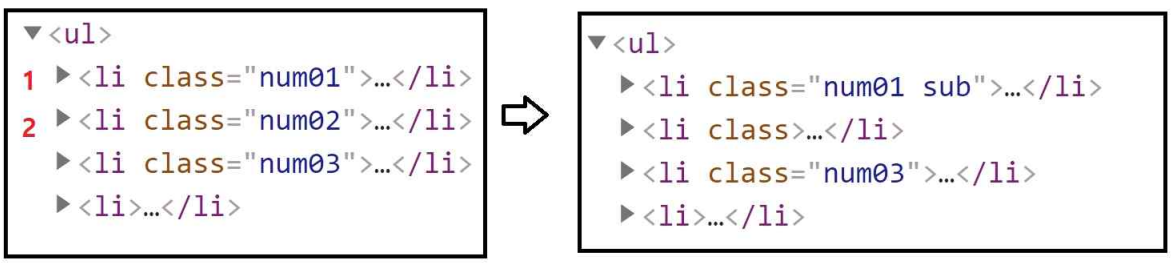
8-3 클래스(classList) 이용하기

프로그램을 사용하면 클래스 속성을 이용하여 작업하는 경우를 많이 봅니다. 아마도 id를 사용하다보면 한 HTML 문서에 중첩을 사용할 수 없으므로 class를 사용하는 작업 패턴이 많이 있습니다. 이번에는 이 클래스 속성을 이용하여 작업할 수 있는 메서드를 알아봅시다.

메서드	설명
classList.add('클래스명')	클래스명을 추가한다.
classList.remove('클래스명')	클래스명을 삭제한다.
classList.toggle('클래스명')	클래스명이 있는 경우에는 없애고, 없는 경우에는 추가한다.
classList.contains('클래스명') has	해당되는 클래스명이 있을 경우 true값을 없을 경우 false를 출력한다.

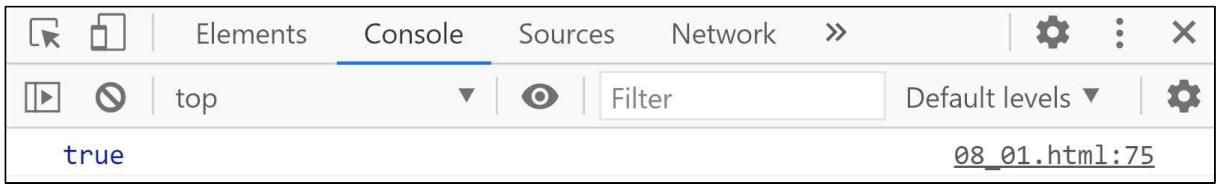
nav>ul>li:nth-child(1)과 nav>ul>li:nth-child(2)에 각각 클래스명을 추가, 삭제요청을 했습니다.

```
71    document.querySelector('nav>ul>li:nth-child(1)').classList.add('sub');❶
72    document.querySelector('nav>ul>li:nth-child(2)').classList.remove('num02');❷
```



로 변경되었습니다. 또 변수 sub에 ul>li:nth-child(3)에 num03클래스가 있는지 확인해 보면

```
74    const sub =
      document.querySelector('nav>ul>li:nth-child(3)').classList.contains('num03');
75    console.log(sub);
```



true값이 출력이 되었습니다.

8-4 DOM의 속성 변경하기

이번에는 선택한 DOM의 요소의 속성노드에 접근하여 변경하는 방법을 알아보겠습니다.

메서드	설명
getAttribute('속성명')	선택된 속성노드의 값을 가져온다.
setAttribute('속성명','값')	선택된 속성노드의 값을 변경한다.
removeAttribute('속성명')	선택된 속성노드의 값을 없앤다.
hasAttribute('속성명')	해당되는 속성명이 있으면 true값을 없으면 false를 출력한다.

8-5 HTML 요소 생성하기

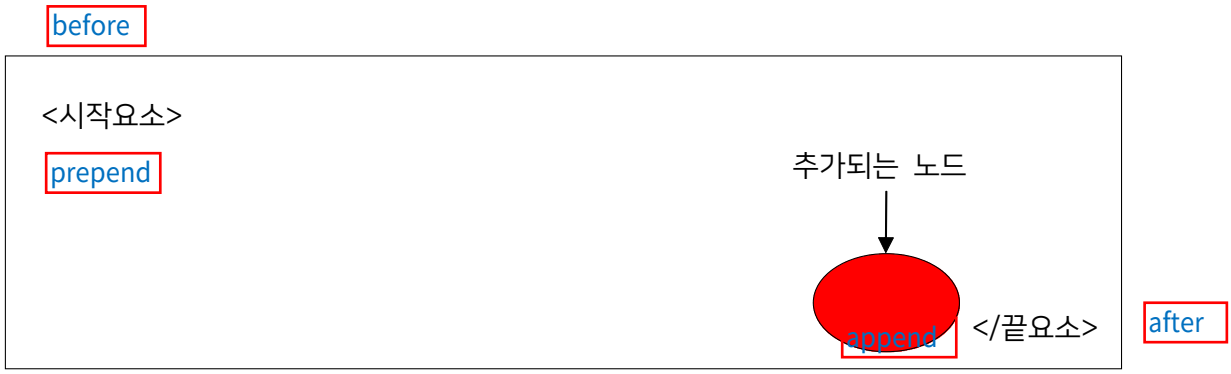
메모리에 생성됨 -> 우리의 눈에 보이기 위해선 후처리가 필요함
- ctrl + c 이후 ctrl + v를 해야 우리 눈에 콘텐츠가 보이는 것 처럼

이번에는 DOM의 요소를 필요한 node를 추가하는 방법에 대해 알아봅시다. DOM필요한 요소를 추가하기 위해서는 필요절차를 만들어야 합니다. 우리가 생각하기로는 필요한 요소를 원하는 요소에 추가하면 문제가 없어 보이지만 실제로 작업하면 노드가 생성되지 않을 경우가 많습니다.

메소드	설명
createElement()	요소를 생성한다.
createTextNode()	text콘텐츠를 생성한다.
appendChild(새로운 노드)	새로 생성된 노드를 요소가 끝나기 전에 삽입한다.
append()	노드나 문자열을 node 끝에 삽입합니다.
prepend()	노드나 문자열을 node 맨 앞에 삽입합니다.
before()	노드나 문자열을 node 이전에 삽입합니다.
after()	노드나 문자열을 node 다음에 삽입합니다.
replaceWith	node를 새로운 노드나 문자열로 대체합니다.
cloneNode(옵션)	노드를 복사

true

createElement()와 createTextNode() 는 요소와 text 콘텐츠를 생성합니다. 이것은 현재 브라우저에서 사용하기 위해 생성하여 자바스크립트에 인식을 시켜주는 과정입니다.
appendChild()와 append()는 필요한 노드를 어느부분에 삽입하여 우리의 눈으로 결과물을 확인 할 수 있는지를 알려줍니다. append()와 appendChild()는 선택된 요소가 끝나기 직전에 즉 </요소> 바로전에 넣어 줍니다.



8-6 이벤트 생성하기

이벤트에 대해 알아보도록 합니다. 이벤트란 어떠한 동작이 발생이 되는 것을 의미합니다. 예를 들면 마우스 버튼을 누른다든지, 아니면 윈도우의 사이즈가 작아진다든지, 스크롤바를 움직인다든지 기존의 화면에서 무언가를 발생시키는 것을 우리는 이벤트라 합니다.

자바스크립트에서는 여러 이벤트의 종류가 있습니다.

이벤트를 발생하였을 때 작동하는 실행문을 연결하는 작업을 '이벤트처리기' 또는 '이벤트핸들러'라고 합니다. 보통 이벤트이름에 'on'을 연결하여 사용할 수 있습니다.

만약에 click 이벤트가 발생이 되었다면 onclick이란 이벤트 핸들러가 작동하여 작업을 실행합니다.

마우스 이벤트

마우스 이벤트는 마우스에서 버튼이나 휠 또는 마우스 커서가 어떠한 영역에 침범을 했을 대 발생하는 이벤트입니다.

이벤트핸들러	설명
onclick	마우스로 클릭
ondblClick	마우스를 더블 클릭
ondragDrop	마우스를 드래그하여 끌어주기
onmouseDown	사용자가 마우스 버튼을 눌렀을 때
onmouseMove	마우스를 이동시키는 이벤트
onmouseOut	어떠한 영역에서 마우스가 벗어 날때
onmouseOver	어떠한 영역에서 마우스가 올라갈 때
onmouseUp	사용자가 마우스 버튼을 놓을 때
onmove	마우스를 움직일 때

키보드 이벤트

키보드 이벤트는 키보드의 키가 조작될 때 발생됩니다.

이벤트핸들러	사용법
onkeydown	사용자가 키를 눌렀을 때 발생하는 이벤트
onkeypress	사용자가 키를 눌렀다가 놓았을 때 발생하는 이벤트
onkeyup	사용자가 키를 눌렀다가 떼었을 때 발생하는 이벤트

문서 로딩 이벤트

서버 사이드에서 문서를 가져오거나 문서를 위아래로 스크롤 할 때와 같이 웹문서를 브라우저 창에 보여주는 여러 작동에서 발생되는 이벤트입니다.

이벤트 핸들러	기능 설명
onabort	사용자의 작업을 빠져나오는 이벤트
onerror	이미지나 윈도우를 읽는 도중 에러가 발생할 때
onload	이미지나 문서 등을 로드 할때
onresize	윈도우 크기를 재조정할 때
onunload	문서 종료할때

이벤트핸들러 사용하기

```
function 함수이름(){
실행문 ;
}
DOM의 객체 . 이벤츠핸들러 = 함수이름;
```

addEventLinstner()

addEventLinstner는 하나의 선택자에 여러 이벤트가 작동되도록

먼저 사용 형식을 알아봅시다.

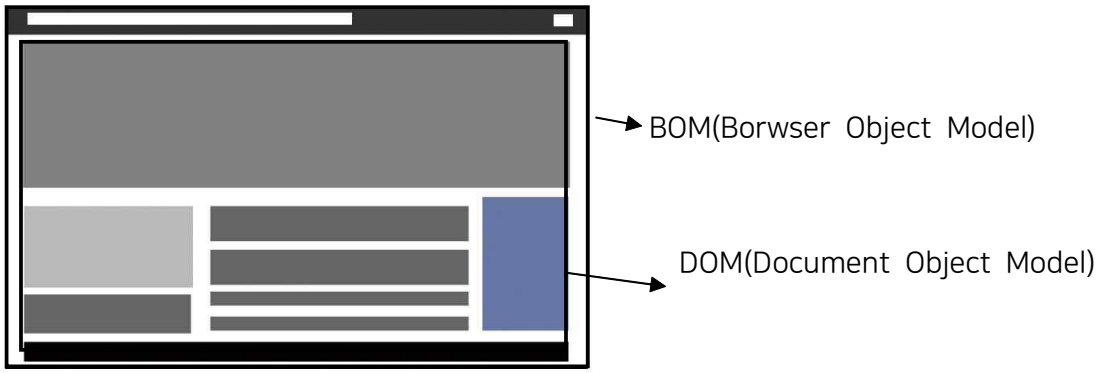
```
function 함수이름(){
실행문 ;
}
DOM의 객체 . addEventListner('이벤트명', 함수이름(익명의 함수))
```

```
55    BTN.addEventListener('click',info);
56    BTN.addEventListener('click',function(){console.log('버튼클릭')})
```

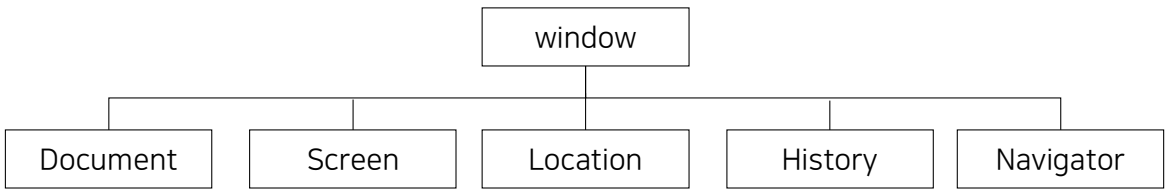
9장 브라우저(BOM)객체 사용하기

9-1 브라우저 객체

브라우저객체란 브라우저에 내장된 객체로 우리는 BOM(Browser Object Model)이라고 합니다. 즉 웹브라우저 전체를 관리하는 시스템을 우리는 브라우저객체라 하고 이 브라우저에 렌더링되어 보여지는 웹 문서를 관리하는 시스템을 도큐멘트 객체(DOM)라 합니다.



브라우저객체의 최상위 객체는 윈도(window)객체 안에는 여러 객체가 존재하며 우리가 8장에서 다룬 DOM도 이 윈도우 객체의 하위 객체입니다.
브라우저의 계층적 구조를 보면 다음과 같습니다.



각 객체에 대해 정리해 보면 다음과 같다.

객체	설명
Window	브라우저 창에 있는 객체로 브라우저 안에 있는 모든 요소의 최상위 객체로 대부분 객체명은 생략하여 사용한다.
Docuement	웹문서 즉 html 문서를 만나면 만들어지는 객체
History	웹브라우저의 방문기록을 저장하는 객체.
Location	웹브라우저의 URL정보를 다루는 객체
Navigator	현재 웹브라우저 정보를 가지고 있는 객체
Screen	브라우저에 보이는 화면 정보에 대한 객체

9-2 window객체

window객체는 브라우저의 최상의 객체며 기본이 되는 객체이다.

window객체 메서드

메서드	설명
alert()	경고용 대화상자를 보여줌
confirm()	확인, 취소를 선택할 수 있는 대화상자를 보여줌
prompt()	입력창이 있는 대화상자를 보여줌
setTimeout()	일정 간격 으로 함수를 호출하여 실행한다. millisecond 단위로 지정
clearTimeout()	setTimeout 메소드를 정지
setInterval()	일정시간마다 함수를 호출하여 실행한다. millisecond 단위로 지정
clearInterval()	setInterval 메소드의 정지
resizeTo()	새로운 창의 크기를 변경한다.
scrollBy()	창을 상대적인 좌표로 스크롤. 창의 표시 영역의 수평방향과 수직방향에 대해 픽셀로 지 정
scroll()	창을 스크롤 한다.
moveTo()	새창의 위치를 이동한다.

타이머 사용하기

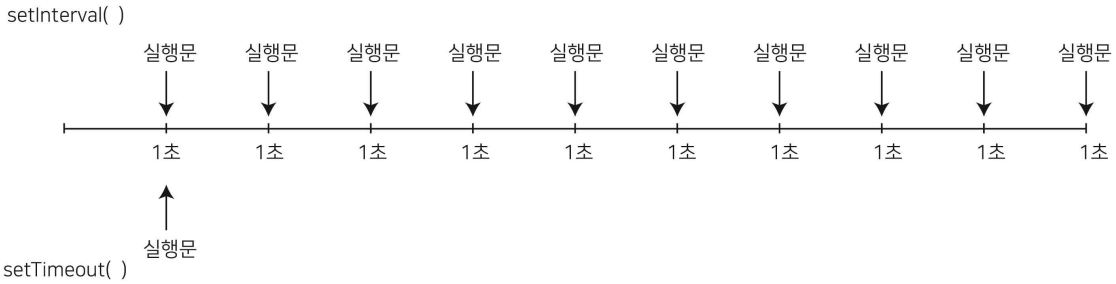
원하는 작업을 일정한 간격을 두고 실행하는 작업을 우리는 타이머라고 합니다. 이러한 타이머는

setTimout() 또는 setInterval()메서드를 사용하여 만들 수 있습니다.

두 메서드의 차이를 먼저 알아봅시다.

먼저 setTimeout()은 타이머를 지정된 시간에 한번만 작동을 시킬 수 있고, setInterval()은 지 일정한 시간을 반복적으로 타이머를 작동 시킬 수 있습니다.

만약에 타이머 작업 필요없게 될 경우에는 타이머를 정지 시켜야 되는데 setTimeout()은 clearTimeout()으로 setInterval()은 clearInterval()로 정지 할 수 있습니다.



1. setInterval( ) , clearInterval( ) 사용하기

setInterval()은 일정한 시간을 반복적으로 원하는 작업을 실행하고자 할 때 사용되면 이러한 setInterval()작업을 취소할 때 clearInterval()을 사용합니다.

먼저 형식을 살펴봅시다.

setInterval()설정하기

```
기본형:,
function 함수명( ){실행문 ;}
❶ var 변수명 = setInterval( 함수명 , 시간간격 );
❷ var 변수명 = setInterval( '함수명( )' , 시간간격 );
```

❶ setInterval()은 window객체의 메서드 이므로 window객체명은 생략을 합니다.

또 타이머에 실행문은 이름 있는 함수를 사용하여 타이머에 실행문을 등록합니다.

❷ setInterval() 설정시 '함수명( )'의 형식을 할 경우에는 따옴표(' ')를 사용해야 합니다.

```
❸ var 변수명 = setInterval( 익명의 함수(function(){실행문} , 시간간격 );
```

❸ 타이머의 실해문을 함수로 연결 하지 않고 직접 실행문을 설정할 경우에는 실행문의 그룹을 이름없는 함수의 형태로 설정해야 합니다.

2. setTimeout() ,clearTimeout() 사용하기

setTimeout()은 일정한 시간이 지난 후 원하는 작업을 실행하고자 할 때 사용됩니다. 이러한 setTimeout()작업을 취소할 때 clearTimeout()을 사용합니다.

먼저 형식을 살펴봅시다.

```
기본형:,
function 함수명( ){실행문 ;}
var 변수명 = setTimeout( 함수명 , 시간간격 );

타이머 중지:
clearTimeout(변수명)
```

9-3 여러 BOM 객체 사용하기

윈도우객체 이외에 사용할 여러 객체를 정리해 봅시다.

스크린객체

스크린객체는 사용자의 모니터에 관한 정보를 제공하는 객체입니다. 모니터의 해상도, 화면 크기, 색상 등에 관련한 정보를 제공하며 사용자 해상도에 따라 적절한 처리가 필요한 경우에 사용합니다.

속성	설명
height	전체 화면의 세로의 크기
width	전체 화면의 가로의 크기
availHeight	작업표시줄을 제외한 화면의 가로크기
availWidth	작업표시줄을 제외한 화면의 세로크기
colorDepth	화면의 색상 수 정보

location객체

location객체는 사용자가 사용하는 웹 브라우저의 주소 입력줄의 주소를 관리하는 객체로 현재의 URL에 대한 정보를 저장하기 위하여 사용합니다.

메서드

메소드	기능 설명
toString()	location.href를 반환합니다.
assign()	location.href를 설정합니다.
reload()	문서를 새로 고치거나 다시 읽게 해 준다.
replace()	현재 문서를 URL 문서로 대체한다.

속성

속성	기능 설명
hash	표식 이름을 설정하거나 검색
host	url 주소의 호스트 이름과 포트를 설정하거나 검색
hostname	url 주소의 호스트 이름이나 호스트의 ip 주소 설정 및 검색
href	문서의 url 주소를 설정하거나 검색
pathname	문서의 디렉토리 위치를 설정하거나 검색
port	포트번호를 설정하거나 검색
protocol	프로토콜 종류를 설정하거나 검색
search	검색엔진을 호출할 때 사용

```
17        var local = location.href;
18        console.log(local);
```

history객체

history 객체는 사용자가 방문했던 URL 리스트 정보를 저장해 두는 곳으로서 히스토리 리스트를 조작하는 기능을 제공 합니다.

메소드	기능 설명
back()	현재의 웹 브라우저에서 방문한 이전 페이지로 이동한다.
forward()	현재의 웹 브라우저에서 방문한 다음 페이지로 이동한다.
go()	상대 숫자를 설정하여 방문한 페이지로 이동한다.

속성	기능 설명
current	현재 문서의 URL 정보
length	history 객체의 주체에 대한 길이 정보
next	history 객체에서 다음 문서의 URL 정보
previous	history 객체에서 이전 문서의 URL 정보
search	검색 문자

```
window.history.go(-2);
```

navigator객체

navigator객체는 웹사이트의 방문자가 사용하는 부라우저 정보와 운영체제 정보를 제공하는 객체입니다.

속성	설명
appCodeName	현재 브라우저의 코드명의 정보를 반환함
appName	현재 브라우저의 이름의 정보를 반환함
appVersion	현재 브라우저의 버전의 정보를 반환함
language	현재 브라우저가 사용하는 언어정보를 반환함
product	현재 브라우저의 엔진 정보를 반환함
platform	현재 브라우저의 운영체제 정보를 반환함
onLine	현재 브라우저의 온라인 상태 정보를 반환함
userAgent	현재 브라우저의 운영체제의 종합 정보를 반환함

```
var NaviInfo = "User-agent header sent: " + navigator.userAgent;
console.log(NaviInfo);
```