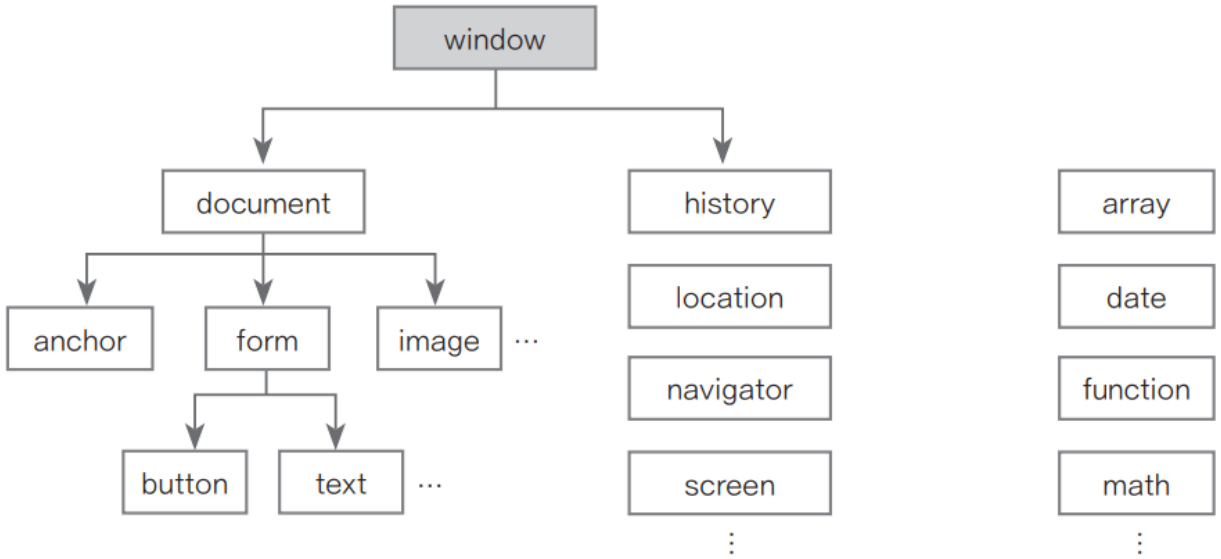
BOM ,DOM

최상위 객체, window

내장 객체란

- 사용자가 가져다 쓸 수 있도록 미리 만들어진 객체.
- 웹 브라우저에 웹 문서가 열리면 가장 먼저 window 객체가 만들어지고 하위에 웹 브라우저 요소에 해당하는 객체들이 만들어진다.
- window 객체를 비롯해 하위에 연결된 객체들은 모두 HTML 웹 API에 만들어진 객체들



window 객체

- Window 객체는 웹 브라우저 창의 상태를 제어하는 객체
- 자바스크립트 객체 중 최상위이자 기본이 되는 객체
- 자바스크립트의 모든 객체는 Window 객체에 포함된다
- Window 객체에는 웹 브라우저 창과 관련된 여러 가지 속성이 있다.
- Window 객체의 속성과 메서드에 접근하는 마침표(.)를 사용한다.

속성	설명
document	브라우저 창에 표시된 웹 문서에 접근할 수 있습니다.
frameElement	현재 창이 다른 요소 안에 포함되어 있으면 그 요소를 반환합니다. 포함되어 있지 않으면 null을 반환합니다.
innerHeight	내용 영역의 높이를 나타냅니다.
innerWidth	내용 영역의 너비를 나타냅니다.
localStorage	웹 브라우저에서 데이터를 저장하는 로컬 스토리지를 반환합니다.
location	Window 객체의 위치/현재 URL을 나타냅니다.
name	브라우저 창의 이름을 가져오거나 수정합니다.
outerHeight	브라우저 창의 바깥 높이를 나타냅니다.
outerWidth	브라우저 창의 바깥 너비를 나타냅니다.

pageXOffset	스크롤했을 때 화면이 수평으로 이동하는 픽셀 수. scrollX와 같습니다.
pageYOffset	스크롤했을 때 화면이 수직으로 이동하는 픽셀 수. scrollY와 같습니다.
parent	현재 창이나 서브 프레임의 부모 프레임입니다.
screenX	브라우저 창의 왼쪽 테두리가 모니터 왼쪽 테두리에서부터 떨어져 있는 거리를 니다냅니다.
screenY	브라우저 창의 위쪽 테두리가 모니터 위쪽 테두리에서부터 떨어져 있는 거리를 나타냅니다.
scrollX	스크롤했을 때 수평으로 이동하는 픽셀 수를 나타냅니다.
scrollY	스크롤했을 때 수직으로 이동하는 픽셀 수를 나타냅니다.
sessionStorage	웹 브라우저에서 데이터를 저장하는 세션 스토리지를 반환합니다.

window 객체의 주요 메서드(함수)

- 대화 창을 표시하거나 브라우저 창의 크기나 위치를 알아내고 지정하는 등
- 웹 브라우저 창 자체와 관련된 것이 대부분
- 앞에서 사용했던 alert( ) 함수나 prompt( ) 함수도 window 객체의 함수
- 객체의 함수 표기법에 따르면 window.alert( )라고 입력해야 하지만, window.를 생략할 수도 있다.

함수	설명
alert( )	알림 창(Alert Dialog)을 표시합니다.
blur( )	창에서 포커스를 제거합니다.
close( )	현재 열려 있는 창을 닫습니다.
confirm( )	[확인], [취소]가 있는 확인 창을 표시합니다.
focus( )	현재 창에 포커스를 부여합니다.
moveBy( )	현재 창을 지정한 크기만큼 이동합니다.
moveTo( )	현재 창을 지정한 좌표로 이동합니다.
open( )	새로운 창을 엽니다.
postMessage( )	다른 창으로 메시지를 전달합니다.
print( )	현재 문서를 인쇄합니다.
prompt( )	프롬프트 창에 입력한 텍스트를 반환합니다.

resizeBy( )	지정한 크기만큼 현재 창 크기를 조절합니다.
resizeTo( )	동적으로 브라우저 창의 크기를 조절합니다.
scroll( )	문서에서 특정 위치로 스크롤합니다.
scrollBy( )	지정한 크기만큼씩 스크롤합니다.
scrollTo( )	지정한 위치까지 스크롤합니다.
setCursor( )	현재 창의 커서를 변경합니다.
showModalDialog( )	모달 창을 표시합니다.
sizeToContent( )	내용에 맞게 창 크기를 맞춥니다.
stop( )	로딩을 중지합니다.

팝업 창을 여는 open() 함수

open( ) 함수를 사용하면 현재 창이나 새 탭, 새로운 알림 창 등 다양한 형태로 새 창을 열 수 있다.

```
window.open(경로, 창 이름, 창 옵션)
```

- 경로: 팝업 창에 표시할 문서나 사이트의 경로(주소).
- 창 이름: 팝업 창의 이름. 이름을 지정하지 않으면 팝업 창이 계속 새로 나타난다.
- 창 옵션: left, top 속성을 사용해 위치를 정하거나 width, height 속성을 사용해 크기를 지정할 수 있다.
  (위치를 지정하지 않으면 팝업 창은 화면의 맨 왼쪽 위에 나타남)

브라우저 창을 닫는 close() 함수

현재 열려 있는 브라우저 창을 닫는 함수

```
window.close()
```

08₩notice.html에 있는 '닫기' 버튼 수정하기

```
<button>닫기</button>
```



```
<button onclick="window.close()">닫기</button>
```

screen 객체

사용자의 화면 크기나 방향 등의 정보를 담고 있는 객체

구분		설명
프로퍼티	availHeight	UI 영역(예를 들어 윈도우의 작업 표시줄이나 Mac의 독)을 제외한 영역의 높이를 나타냅니다.
	availWidth	UI 영역을 제외한 내용 표시 영역의 너비를 나타냅니다.
	colorDepth	화면에서 픽셀을 렌더링할 때 사용하는 색상 수를 나타냅니다.
	height	UI 영역을 포함한 화면의 높이를 나타냅니다.
	orientation	화면의 현재 방향을 나타냅니다.
	pixelDepth	화면에서 픽셀을 렌더링할 때 사용하는 비트 수를 나타냅니다.
	width	UI 영역을 포함한 화면의 너비를 나타냅니다.
메서드	lockOrientation()	화면 방향을 잠급니다.
	unlockOrientation()	화면 방향 잠금을 해제합니다.

history 객체

- history 객체에는 브라우저에서 [뒤로]나 [앞으로] 또는
  주소 표시줄에 입력해서 방문한 사이트 주소가 배열 형태로 저장된다.
- 브라우저 히스토리는 보안 문제 때문에 읽기 전용

구분		설명
프로퍼티	length	현재 브라우저 창의 history 목록에 있는 항목의 개수, 즉 방문한 사이트 개수가 저장됩니다.
메서드	back()	history 목록에서 이전 페이지를 현재 화면으로 불러옵니다
	forward()	history 목록에서 다음 페이지를 현재 화면으로 불러옵니다
	go()	history 목록에서 현재 페이지를 기준으로 상대적인 위치에 있는 페이지를 현재 화면으로 불러옵니다. 예를 들어 history.go(1)은 다음 페이지를 가져오고, history.go(-1)은 이전 페이지를 불러옵니다.

location 객체

현재 문서의 URL 주소와 관련된 정보가 들어 있음

구분		설명
프로퍼티	hash	URL 중에서 #로 시작하는 해시 부분의 정보를 담고 있습니다.
	host	URL의 호스트 이름과 포트 번호를 담고 있습니다.
	hostname	URL의 호스트 이름이 저장됩니다.
	href	전체 URL입니다. 이 값을 변경하면 해당 주소로 이동할 수 있습니다.
	pathname	URL 경로가 저장됩니다.
	port	URL의 포트 번호를 담고 있습니다.
	protocol	URL의 프로토콜을 저장합니다.
	password	도메인 이름 앞에 username과 password를 함께 입력해서 접속하는 사이트의 URL일 경우에 password 정보를 저장합니다.
	search	URL 중에서 ?로 시작하는 검색 내용을 저장합니다.
	username	도메인 이름 앞에 username을 함께 입력해서 접속하는 사이트의 URL일 경우에 username 정보를 저장합니다.
메서드	assign()	현재 문서에 새 문서 주소를 할당해서 새 문서를 가져옵니다.
	reload()	현재 문서를 다시 불러옵니다.
	replace()	현재 문서의 URL을 지우고 다른 URL의 문서로 교체합니다.
	toString()	현재 문서의 URL을 문자열로 반환합니다.

DOM과 DOM 트리

문서 객체 모델(DOM)이란

- 자바스크립트를 이용하여 웹 문서에 접근하고 제어할 수 있도록
  객체를 사용해 웹 문서를 체계적으로 정리하는 방법
- 웹 문서를 구조화한 DOM 트리(DOM tree)와 이벤트 등을 정리해 놓은 표준

웹에서 자바스크립트를 사용하는 이유는 어떤 조건이 주어지거나 사용자의 동작이 있을 때 웹 문서 전체 또는 일부분이 동적으로 반응하게 하는 것 → 이렇게 하려면 웹 문 서의 모든 요소를 따로 제어할 수 있어야 한다.

(예) 웹 문서에 텍스트와 이미지가 들어 있다면

- 웹 브라우저는 마크업 정보를 보면서 텍스트 단락이 몇 개이고 그 내용이 무엇인지 살펴본다
- 이미지가 몇 개이고 이미지 파일 경로는 어떠한지 대체 텍스트는 무엇인지도 파악해서 이미지별로 정리해서 인식한다.
- 텍스트와 이미지 요소를 브라우저가 제어하려면 두 요소를 따로 구별해서 인식해야 한다.
- 마크업을 보면서 요소 사이의 포함 관계도 알아야 한다.

DOM 트리와 노드

DOM에는 단순히 태그에 해당하는 요소 노드뿐만 아니라 여러 종류의 노드가 있다.
DOM 트리에서 가지가 갈라져 나가는 부분은 노드라고 하고,
DOM 트리의 시작 부분, 즉 html 노드를 나무의 뿌리에 해당하는 루트 노드라고 한다.

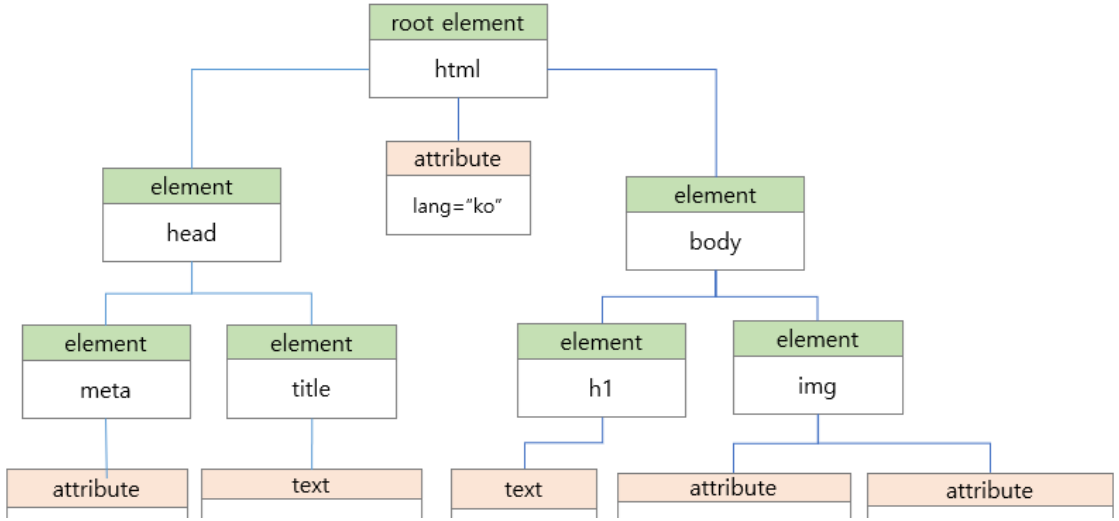
노드를 구성하는 원칙

- 모든 HTML 태그는 **요소 노드**가 된다.
- HTML 태그에서 사용하는 텍스트 내용은 자식 노드인 **텍스트 노드**가 된다.
- HTML 태그에 있는 속성은 모두 자식 노드인 **속성 노드**가 된다.
- 주석들은 **주석 노드**가 된다.

DOM 트리와 노드

- DOM은 문서의 요소 뿐만 아니라 각 요소의 내용과 속성도 자식으로 나타냄

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset=“utf-8”>
  <title>DOM Tree 알아보기</title>
</head>
<body>
  <h1>제목</h1>
  <img src="images/cat.jpg"  alt=" 고양이 이미지">
</body>
</html>
```



노드 리스트

- querySelectorAll() 메서드를 사용하면 여러 개의 노드를 한꺼번에 가져올 수 있다.
- 가져온 여러 개의 노드 정보를 저장한 것을 노드 리스트라고 한다.
- 노드 리스트는 배열과 비슷하게 생겼고 배열처럼 사용할 수 있다. (배열은 아님)

노드 리스트

- querySelectorAll() 메서드를 사용하면 여러 개의 노드를 한꺼번에 가져올 수 있다.
- 가져온 여러 개의 노드 정보를 저장한 것을 노드 리스트라고 한다.
- 노드 리스트는 배열과 비슷하게 생겼고 배열처럼 사용할 수 있다. (배열은 아님)

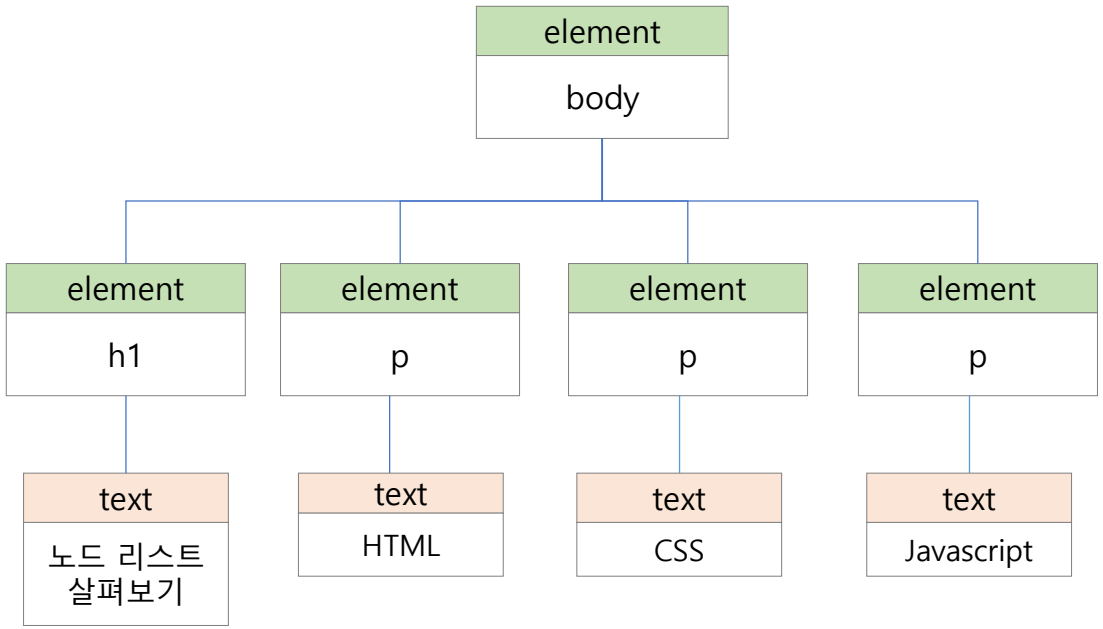
```
<body>
  <h1>노드 리스트 살펴보기</h1>
  <p>HTML</p>
  <p>CSS</p>
  <p>Javascript</p>
</body>
```

노드 리스트 살펴보기

HTML

CSS

Javascript



웹 요소에 접근하기

웹 요소에 접근하기

웹 문서에서 원하는 요소를 찾아가는 것을 “접근한다(access)”고 함

(예) 제목의 글자색을 바꾸고 싶다면 우선 제목까지 ‘접근해야’ 함.

querySelector(), querySelectorAll() 함수

querySelector(선택자)	querySelectorAll(선택자)

선택자

id 이름 앞에는 해시 기호(#), class 이름 앞에는 마침표(.), 태그는 기호 없이 태그명 사용

반환 값

- querySelecotor( ) 메서드는 한 개의 값만 반환
- querySelecotorAll( ) 메서드는 반환 값이 여러 개일 때 모두 반환 → 노드 리스트로 저장됨

getElement~ 함수

- 예전부터 사용하던 방법
- id나 class, 태그명을 사용해서 접근

getElementById( )

```
document.getElementById(“id명”)
```

getElementsByClassName( )

```
document.getElementsByClassName(“클래스명”)
```

getElementsByTagName( )

```
document.getElementsByTagName(“태그명”)
```

getElement*() 와 querySelector()의 차이

- id값이나 class값, 태그 이름을 사용해서 접근하는 것은 같다
- querySelector()를 사용하면 둘 이상의 선택자를 조합해서 접근할 수 있음 (예, #detail > p)

웹 요소 내용 가져오기 및 수정하기

접근한 요소의 텍스트 내용을 가져오거나 지정할 때는 innerText, innerHTML, textContent 프로퍼티 사용

- innerText : 순수 텍스트를 가져오거나, 해당 요소에 텍스트 지정
- innerHTML : 태그와 함께 텍스트를 가져오거나, 해당 요소에 태그와 함께 텍스트 지정
- textContent : 텍스트를 가져오되, 화면에 보이는대로가 아니라 소스에 있는대로 가져옴
  (화면에서 감춘 요소에서도 내용을 가져올 수 있고, 소스에 공백이 여러 개일 경우 그 공백도 모두 가져옴)

```
요소.innerText
요소.innerHTML
요소.textContent
```

```
요소.innerText = "내용"
요소.innerHTML = "내용"
요소.textContent = "내용"
```

자바스크립트로 스타일 수정하기

CSS 속성에 접근하기

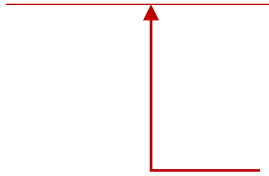
자바스크립트를 이용하면 스타일 속성의 값을 가져오거나 원하는 값으로 수정할 수 있습니다.

→ 웹 문서에서 다양한 효과를 만들 수 있습니다.

```
요소.style.속성명
```

예) 글자색 수정하려면 style.color

배경색 수정하려면 style.backgroundColor



background-color, font-size 처럼 하이픈을 사용해 두 단어를 연결한 CSS 속성

→ backgroundColor, fontSize 처럼 하이픈 없이 사용. 둘째 단어는 대문자로

classList 프로퍼티

- 두 개 이상의 class 스타일이 적용되었을 경우 class 스타일 정보를 담아두는 프로퍼티.
- classList를 사용해서 적용 중인 class 스타일을 제거할 수도 있고 새로운 class 스타일을 추가할 수도 있다

05₩classlist-0.html

```
<div id="desc">
  <p class="user clicked">이름 : 도레미</p>
  <p class="user">주소 : somewhere</p>
  <p class="user">연락처 : 1234-5678</p>
</div>
```

콘솔 창에서 확인하기

```
document.querySelector("#desc p").classList
```

클래스 스타일 추가하기 및 삭제하기

새로운 클래스 스타일을 추가하거나 (이 경우, 클래스 스타일이 만들어져 있어야 함)
기존에 적용 중인 클래스 스타일을 제거할 수 있습니다.

```
요소.classList.add(클래스명)
요소.classList.remove(클래스명)
```

contains() – 특정 클래스 스타일 있는지

앞의 예제에서 제목을 클릭하면 배경색과 글자색이 바뀌는데, 원래 상태로 되돌아가지는 않는다.

추가했던 클래스 스타일을 제거하면 원래대로 돌아간다

그러기 위해서는 특정 클래스 스타일이 있는지 체크할 수 있어야 한다

→ classList의 contains() 함수 사용

```
요소.classList.contains(클래스명)
```

toggle() – 특정 스타일 토글

특정 클래스를 추가하거나 삭제하기를 반복할 경우에는 `classList`의 `toggle()` 함수가 더 편리하다.

```
요소.classList.toggle(클래스명)
```