

# 17. 문서 객체 모델(DOM)

17-1 문서 객체 모델 알아보기

17-2 DOM 요소에 접근하고 속성 가져오기

17-3 DOM에서 이벤트 처리하기

17-4 DOM에서 노드 추가, 삭제하기



# 문서 객체 모델 알아보기

## 문서 객체 모델이란

자바스크립트를 이용하여 웹 문서에 접근하고 제어할 수 있도록 객체를 사용해 웹 문서를 체계적으로 정리하는 방법

웹 문서와 그 안의 모든 요소를 '객체'로 인식하고 처리함

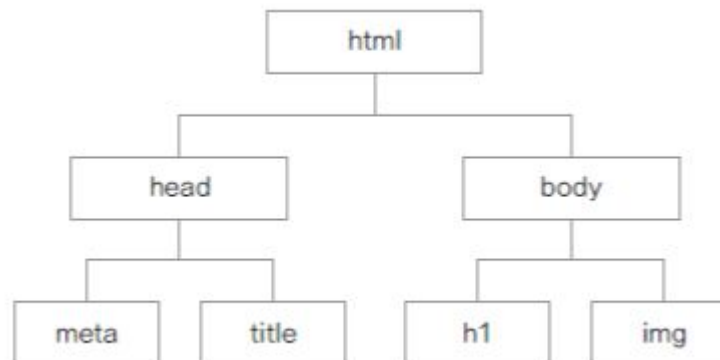
예) 웹 문서 전체는 document 객체, 삽입한 이미지는 image 객체



### Do it! HTML의 요소 관계 알아보기

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>DOM Tree 알아보기</title>
</head>
<body>
  <h1>Do it!</h1>
  
</body>
</html>
```

### 요소의 계층 관계



# 문서 객체 모델 알아보기

## DOM 트리

- 웹 문서에 있는 요소들 간의 부모, 자식 관계를 계층 구조로 표시한 것
- 나무 형태가 되기 때문에 “DOM 트리”라고 함.
- 노드(node) : DOM 트리에서 가지가 갈라져 나간 항목
- 루트 노트(root node) : DOM 트리의 시작 부분(html)

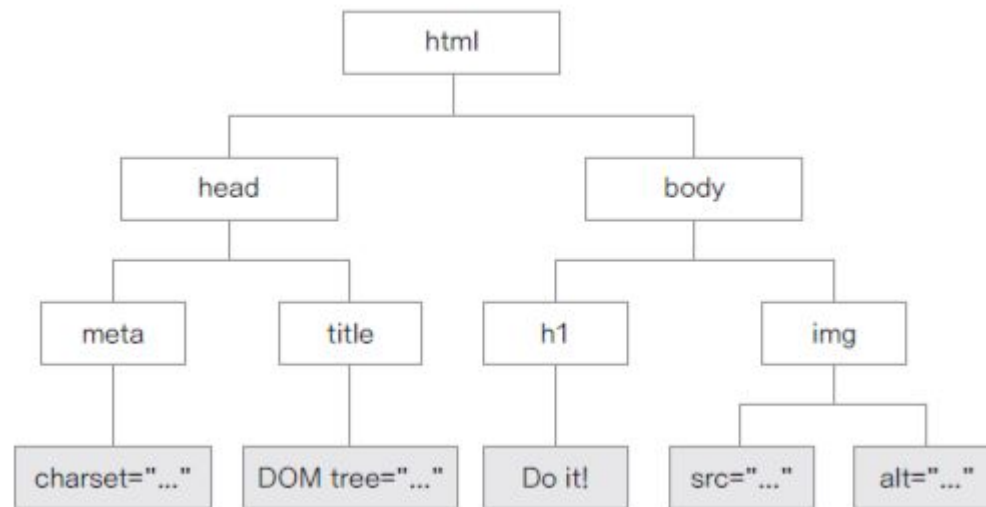


### Do it! HTML의 요소 관계 알아보기

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>DOM Tree 알아보기</title>
</head>
<body>
  <h1>Do it!</h1>
  
</body>
</html>
```

## DOM 을 구성하는 원칙

- 모든 HTML 태그는 요소(element) 노드
- 웹 문서의 텍스트 내용은 요소 노드의 자식 노드인 텍스트(text) 노드
- 태그의 속성은 요소 노드의 자식 노드인 속성(attribute) 노드
- 주석은 주석(comment) 노드

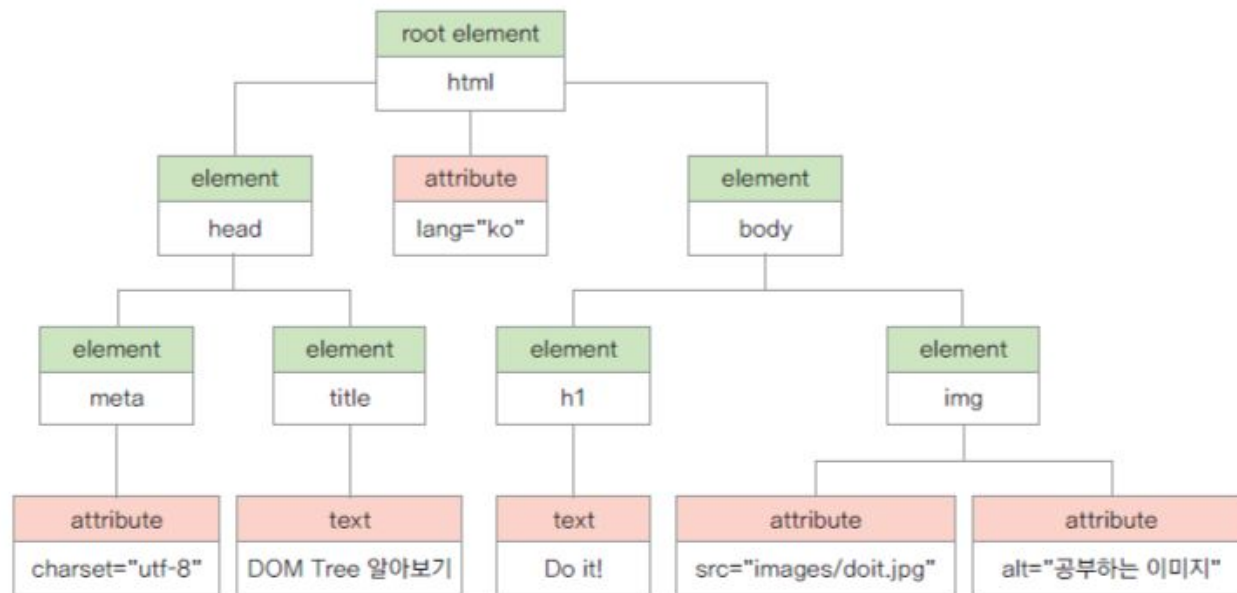


# 문서 객체 모델 알아보기



## Do it! HTML의 요소 관계 알아보기

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>DOM Tree 알아보기</title>
</head>
<body>
  <h1>Do it!</h1>
  
</body>
</html>
```



# DOM 요소에 접근하고 속성 가져오기

## DOM 요소에 접근하기

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

### getElementById( ) 메서드

기본형 요소명.getElementById("id명")

### getElementsByClassName( ) 메서드

기본형 요소명.getElementsByClassName("class명")

### getElementsByTagName( )

#### 메서드

기본형 요소명.getElementsByTagName("태그명")

- 반환 값이 2개 이상일 수 있음
- HTMLCollection 객체에 저장됨

### querySelector( ) 메서드, querySelectorAll( ) 메서드

기본형 노드.querySelector(선택자)  
노드.querySelectorAll(선택자 또는 태그)

- querySelector( ) 메서드는 한 개의 값만 반환
- querySelectorAll( ) 메서드는 반환 값이 여러 개일 때 모두 반환 □ 노드 리스트로 저장됨
- id 이름 앞에는 해시 기호(#), class 이름 앞에는 마침표(.), 태그는 기호 없이 태그명 사용

# DOM 요소에 접근하고 속성 가져오기

## innerText, innerHTML 프로퍼티

웹 요소의 내용을 수정하는 프로퍼티

- innerText : 텍스트 내용 지정
- innerHTML : HTML 태그까지 포함해서 텍스트 내용 지정

기본형 요소명.innerText = 내용  
요소명.innerHTML = 내용

```
<button onclick="inntext()">innerText로 표시하기</button>
<button onclick="innhtml()">innerHTML로 표시하기</button>
<h1>현재 시각: </h1>
<div id="current"></div>

<script>
  var now = new Date();

  function inntext(){
    document.getElementById("current").innerText = now;
  }
  f      innhtml() {
    document.getElementById("current").i      = "<em>" + now + "</em>";
  }
</script>
```



# DOM 요소에 접근하고 속성 가져오기

## getAttribute( ) 메서드, setAttribute( ) 메서드

- getAttribute( ) 메서드 : 속성 노드의 값을 가져옴
- setAttribute( ) 메서드 : 속성 노드의 값을 바꿈

기본형 `getAttribute("속성명")`

기본형 `setAttribute("속성명", "값")`

```
<div id="prod-pic">
  
  *****
</div>

<script>
function displaySrc() {
  var cup = document.querySelector("#cup");           // id="cup"인 요소에 접근하기
  alert("이미지 소스: " + cup.getAttribute("src"));    // cup 속성을 알림 창에 표시
}
</script>
```



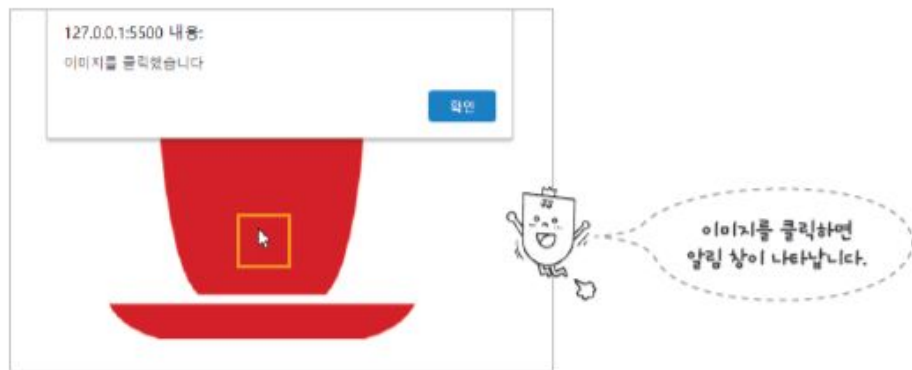


# DOM에서 이벤트 처리하기

## DOM 요소에 함수 직접 연결하기

DOM 요소에 이벤트 처리기 함수를 직접 연결

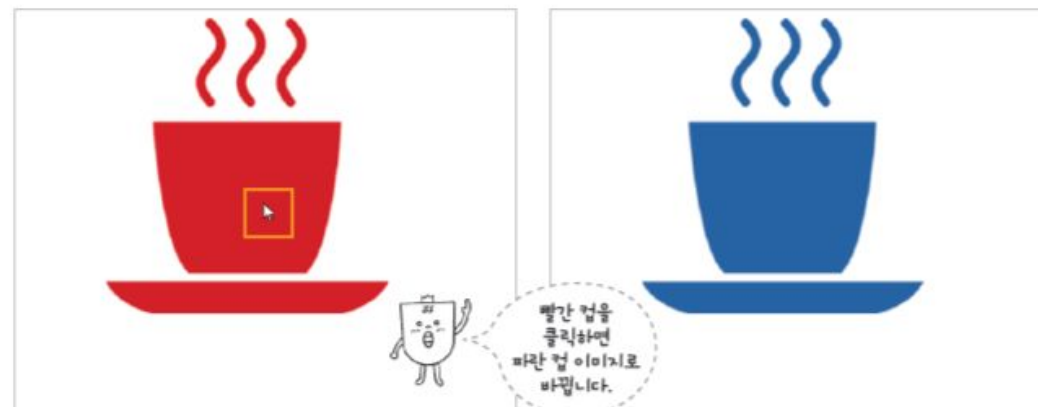
```
(... 생략 ...)  
  
  
.....  
<script>  
  var cup = document.querySelector("#cup");  
  cup.onclick = function(){  
    alert("이미지를 클릭했습니다");  
  }  
</script>
```



## 함수 이름을 사용해 연결하기

- 함수를 따로 정의해 놓았다면 DOM 요소에 함수 이름을 사용해 연결
- 이 때 함수 이름 다음에 괄호를 추가하지 않음

```
  
  
.....  
<script>  
  var cup = document.querySelector("#c"); // id=cup 요소를 가져옴  
  cup.o = changePic; // cup을 클릭하면 ChangPic 함수를 실행  
  
  function changePic() {  
    cup.src = "images/cup-2.png"; // cup 요소의 경로를 다른 이미지 경로로 바꿈  
  }  
</script>
```





# DOM에서 이벤트 처리하기

## DOM의 event 객체

웹 문서에서 이벤트 발생한 요소가 무엇인지,  
어떤 이벤트가 발생했는지 등의 정보가 담긴 객체



**Do it!** 이미지에서 클릭한 위치 알아내기

예제 파일 17\event-3.html

(... 생략 ...)

```

```

.....

```
<script>
```

```
var cup = document.querySelector("#cup");
```

```
cup.onclick = function(event) {
```

```
    alert("이벤트 유형: " + event.type + ", 이벤트 발생 위치: " + event.pageX + ", " +  
event.pageY);
```

```
}
```

```
</script>
```

127.0.0.1:5500 내용:

이벤트 유형: click, 이벤트 발생 위치: 351,211

확인



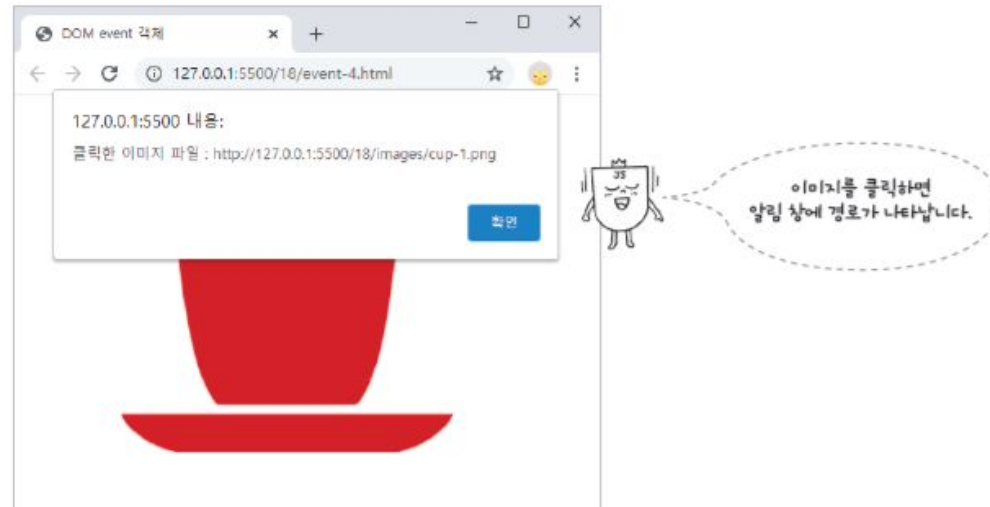
이번에는 이미지를 클릭하면  
알림 창에 클릭한 위치가  
나타납니다.

	구분	설명
프로퍼티	altKey	이벤트가 발생할 때 <b>(Alt)</b> 를 눌렀는지 여부를 boolean값으로 반환합니다.
	button	마우스에서 누른 버튼의 키값을 반환합니다.
	charCode	keypress 이벤트가 발생할 때 어떤 키를 눌렀는지 유니코드값으로 반환합니다.
	clientX	이벤트가 발생한 가로 위치를 반환합니다.
	clientY	이벤트가 발생한 세로 위치를 반환합니다.
	ctrlKey	이벤트가 발생했을 때 <b>(Ctrl)</b> 를 눌렀는지 여부를 boolean값으로 반환합니다.
	pageX	현재 문서 기준으로 이벤트가 발생한 가로 위치를 반환합니다.
	pageY	현재 문서 기준으로 이벤트가 발생한 세로 위치를 반환합니다.
	screenX	현재 화면 기준으로 이벤트가 발생한 가로 위치를 반환합니다.
	screenY	현재 화면 기준으로 이벤트가 발생한 세로 위치를 반환합니다.
	shiftKey	이벤트가 발생할 때 <b>(Shift)</b> 를 눌렀는지 여부를 boolean값으로 반환합니다.
	target	이벤트가 최초로 발생한 대상을 반환합니다.
	timeStamp	이벤트가 발생한 시간을 반환합니다.
	type	발생한 이벤트 이름을 반환합니다.
	which	키보드와 관련된 이벤트가 발생할 때 키의 유니코드값을 반환합니다.
메서드	preventDefault()	이벤트를 취소할 수 있는 경우에 취소합니다.

# DOM에서 이벤트 처리하기

this

이벤트가 발생한 대상에 접근할 때 사용하는 예약어



# DOM에서 이벤트 처리하기

## addEventListener( ) 메서드 사용하기

이벤트 객체를 사용해 이벤트 처리기 연결

기본형 요소.addEventListener(이벤트, 함수, 캡처 여부);

1

2

3

- 1 이벤트: 이벤트 유형을 지정합니다. 단, click과 keypress처럼 on을 붙이지 않고 씁니다.
- 2 함수: 이벤트가 발생하면 실행할 명령이나 함수를 지정합니다. 여기에서 함수를 정의할 때는 event 객체를 인수로 받습니다.
- 3 캡처 여부: 이벤트를 캡처하는지 여부를 지정하며 기본값은 false이고 true와 false 중에서 선택할 수 있습니다. true이면 캡처링, false이면 버블링한다는 의미입니다. 이벤트 캡처링은 DOM의 부모 노드에서 자식 노드로 전달되는 것이고, 이벤트 버블링은 DOM의 자식 노드에서 부모 노드로 전달되는 것입니다.

# DOM에서 이벤트 처리하기

## addEventListener( ) 메서드 사용하기



Do it! 마우스 포인터를 올리면 이미지 바꾸기

예제 파일 17\event-5.html

(... 생략 ...)

```

```

.....

```
<script>
```

```
var cover = document.getElementById("cover");
```

```
cover.addEventListener("mouseover", changePic); // 포인터를 올리면 changePic() 실행
```

```
cover.addEventListener("mouseout", originPic); // 포인터를 내리면 originPic() 실행
```

```
function changePic() {
```

```
    cover.src = "images/easys-2.jpg"; // 이미지 경로를 easys-2.jpg로 바꿈
```

```
}
```

```
function originPic() {
```

```
    cover.src = "images/easys-1.jpg"; // 이미지 경로를 easys-1.jpg로 바꿈
```

```
}
```

```
</script>
```

(... 생략 ...)

메서드 안에서 함수 표현식으로 사용 가능

```

```

```
<script>
```

```
var cover = document.getElementById("cover");
```

```
cover.addEventListener("mouseover", changePic);
```

```
cover.addEventListener("mouseout", originPic);
```

```
function changePic() {
```

```
    cover.src = "images/easys-2.jpg";
```

```
}
```

```
function originPic() {
```


```
    cover.src = "images/easys-1.jpg";
```

```
}
```

```
</script>
```

# DOM에서 이벤트 처리하기

## addEventListener( ) 메서드 사용하기

 **Do it!** 마우스 포인터를 올리면 이미지 바꾸기

예제 파일 17\event-5.html

(... 생략 ...)

```

```

.....

```
<script>
```

```
var cover = document.getElementById("cover");
```

```
cover.addEventListener("mouseover", changePic); // 포인터를 올리면 changePic() 실행
```

```
cover.addEventListener("mouseout", originPic); // 포인터를 내리면 originPic() 실행
```

```
function changePic() {
```

```
    cover.src = "images/easys-2.jpg"; // 이미지 경로를 easys-2.jpg로 바꿈
```

```
}
```


```
function originPic() {
```

```
    cover.src = "images/easys-1.jpg"; // 이미지 경로를 easys-1.jpg로 바꿈
```

```
}
```

```
</script>
```

(... 생략 ...)

 **Do it!** 메서드 안에서 함수 선언하기

예제 파일 17\event-6.html

(... 생략 ...)

```

```

.....

```
<script>
```

```
var cover = document.getElementById("cover");
```

```
cover.addEventListener("mouseover", function() {
```

```
    cover.src = "images/easys-2.jpg";
```

```
});
```

```
cover.addEventListener("mouseout", function() {
```

```
    cover.src = "images/easys-1.jpg";
```

```
});
```

```
</script>
```

(... 생략 ...)

출처: getElementById, addEventListener



왼쪽 이미지 위에  
마우스 포인터를  
올리면 오른쪽  
이미지로  
바뀝니다.


# DOM에서 이벤트 처리하기

## CSS 속성에 접근하기

자바스크립트를 사용해 각 요소의 스타일을 자유롭게 수정할 수 있음

기본형 `document.요소명.style.속성명`

예) id가 desc인 요소의 글자를 파란색으로 변경하려면

 id가 desc인 요소의 글자색 바꾸기

```
document.getElementById("desc").style.color = "blue";
```

- color처럼 한 단어인 속성명은 그대로 사용
- background-color, borderradius처럼 중간에 하이픈(-)이 있는 속성은 backgroundColor나 borderRadius처럼 두 단어를 합쳐서 사용
- 이때 두 번째 단어의 첫 글자는 Color와 Radius처럼 대문자로 표시



Do it! 도형의 테두리와 배경색 바꾸기

예제 파일 17\domCss.html

(... 생략 ...)

```
<div id="rect"></div>
```

.....

```
<script>
```

```
var myRect = document.querySelector("#rect");
```

```
myRect.addEventListener("mouseover", function() { // 마우스 포인터를 올리면
```

```
    myRect.style.backgroundColor = "green";           // 초록색으로 지정
```

```
    myRect.style.borderRadius = "50%";               // 테두리 둥글기를 50%로 지정
```

```
});
```

```
myRect.addEventListener("mouseout", function() { // 마우스 포인터를 내리면
```

```
    myRect.style.backgroundColor = "";              // 배경색 원래 값으로 지정
```

```
    myRect.style.borderRadius = "";                 // 테두리 원래 값으로 지정
```

```
});
```

```
</script>
```

도형 위로 마우스 포인터를 올려놓으세요.



마우스 포인터를  
올리거나 내리면  
도형이 바뀝니다.

도형 위로 마우스 포인터를 올려놓으세요.





# DOM에서 노드 추가, 삭제하기

## 노드 리스트란

querySelectorAll( ) 메서드를 사용해 가져온 여러 개의 노드를 저장한 것

```
(... 생략 ...)  
<h1>Web Programming</h1>  
<ul id="itemList">  
  <li>HTML</li>  
  <li>CSS</li>  
  <li>Javascript</li>  
</ul>
```





# DOM에서 노드 추가, 삭제하기

## 텍스트 노드를 사용하는 새로운 요소 추가하기

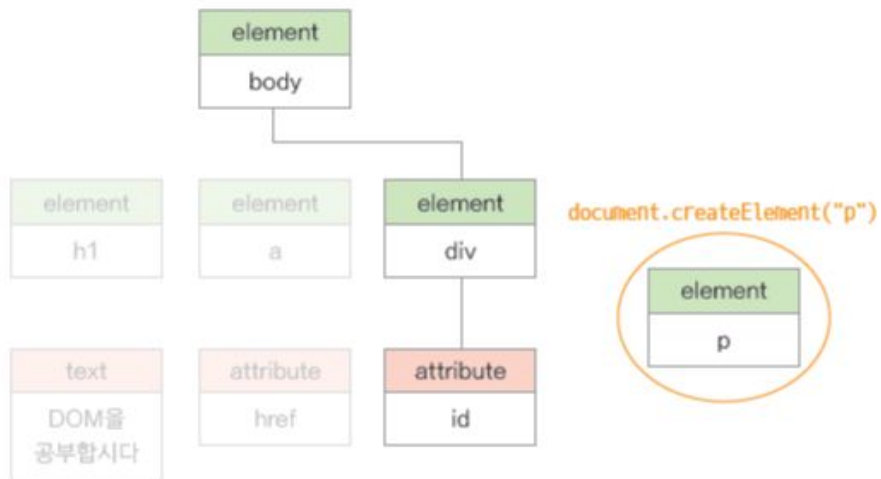
└─ '더 보기' 링크를 클릭하면 텍스트 표시하기

### 1. 요소 노드 만들기 - createElement( ) 메서드

기본형 `document.createElement(노드명)`


 p 요소 노드 만들기

```
var newP = document.createElement("p");
```

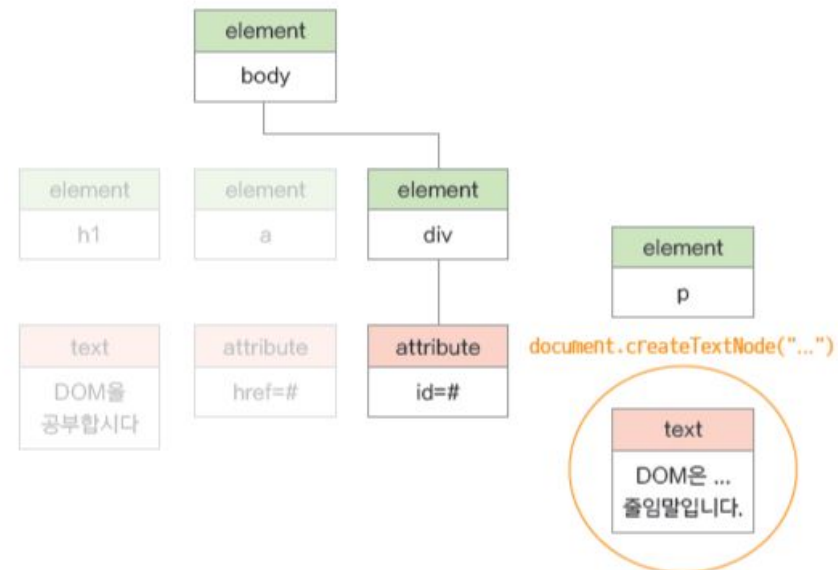


### 2. 텍스트 노드 만들기 - createTextNode( ) 메서드

기본형 `document.createTextNode(텍스트);`

 p 요소의 텍스트 노드 만들기

```
var txtNode = document.createTextNode("DOM은 document object model의 줄임말입니다.");
```



# DOM에서 노드 추가, 삭제하기

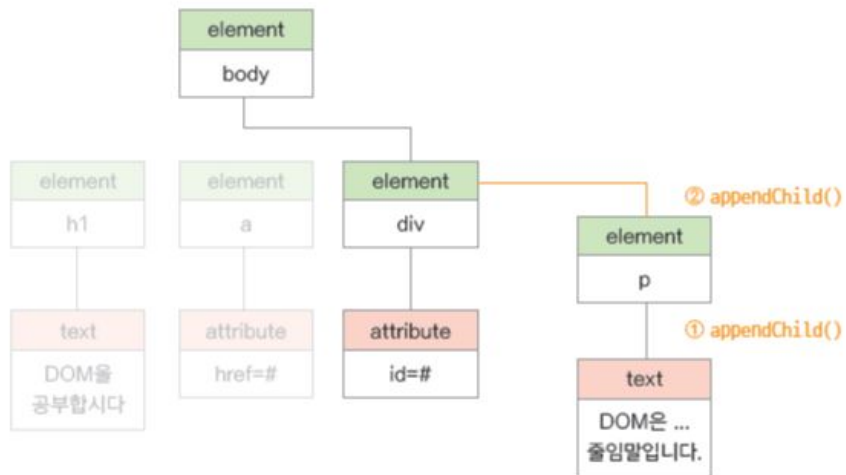
## 텍스트 노드를 사용하는 새로운 요소 추가하기

### 3. 자식 노드 연결하기 - appendChild( ) 메서드

기본형 부모노드.appendChild(자식노드)

텍스트 노드를 자식 노드로 연결하기

```
newP.appendChild(txtNode);  
document.getElementById("info").appendChild(newP);
```



### 전체 소스



링크를 클릭하면 텍스트 표시하기

예제 파일 17\domNode-2.html

(... 생략 ...)

```
<div id="container">  
  <h1>DOM을 공부합니다</h1>  
  <a href="#" onclick="addP(); this.onclick='';">더 보기</a>  
  <div id="info"></div>  
</div>
```

링크를 클릭하면 addP() 함수가 한 번만 실행되도록 합니다.

<script>

```
function addP() {  
  var newP = document.createElement("p");  
  var txtNode = document.createTextNode("DOM은 document object model의 줄임말입니다.");  
  newP.appendChild(txtNode);  
  document.getElementById("info").appendChild(newP);  
}
```

</script>


(... 생략 ...)

# DOM에서 노드 추가, 삭제하기

## 속성 값이 있는 새로운 요소 추가하기

└── '더 보기' 링크를 클릭하면 텍스트와 함께 이미지 표시하기

### 1. 요소 노드 만들기 - createElement( ) 메서드


 이미지 노드 추가하기

```
var newImg = document.createElement("img");
```

### 2. 속성 노드 만들기 - createAttribute( ) 메서드

[기본형]

```
document.createAttribute(속성명)
```

 이미지의 src와 alt 속성 만들고 지정하기

```
var srcNode = document.createAttribute("src");  
var altNode = document.createAttribute("alt");  
srcNode.value = "images/dom.jpg";    // src 속성값 지정  
altNode.value = "돔 트리 예제 이미지"; // alt 속성값 지정
```

### 3. 속성 노드 연결하기 - setAttributeNode( ) 메서드

[기본형]

```
요소명.setAttributeNode(속성노드)
```

 이미지의 src 속성 노드 연결하기

```
newImg.setAttributeNode(srcNode);
```

### 4. 자식 노드 연결하기 - appendChild( ) 메서드

```
document.getElementById("info").appendChild(newImg);
```

# DOM에서 노드 추가, 삭제하기

## 텍스트 노드를 사용하는 새로운 요소 추가하기

전체 소스



Do it! 링크를 클릭하면 텍스트와 이미지 표시하기

예제 파일 17\domNode-3.html

(... 생략 ...)

```
<div id="container">
  <h1>DOM을 공부합시다</h1>
  <a href="#" onclick="addContents(); this.onclick='';">더 보기</a>
  <div id="info"></div>
</div>
<script>
  function addContents() {
    var newP = document.createElement("p");
    var txtNode = document.createTextNode("DOM은 document object model의 줄임말입
니다.");
    newP.appendChild(txtNode);
```

```
    var newImg = document.createElement("img");
    var srcNode = document.createAttribute("src");
    var altNode = document.createAttribute("alt");
    srcNode.value = "images/dom.jpg";
    altNode.value = "돔 트리 예제 이미지";
    newImg.setAttributeNode(srcNode);
    newImg.setAttributeNode(altNode);
```

```
    document.getElementById("info").appendChild(newP);
    document.getElementById("info").appendChild(newImg);
  }
</script>
```

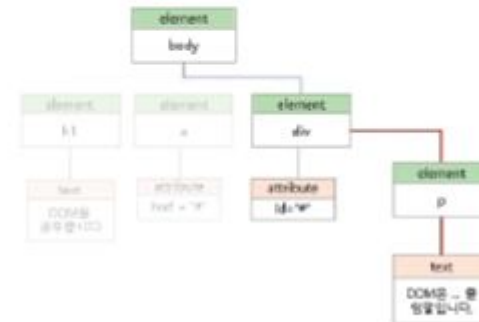
(... 생략 ...)

앞의 예제에서 추가된 부분입니다.

## DOM을 공부합시다



DOM은 document object model의 줄임말입니다.



3.0.1-5500/17/domNode-3.html#

# DOM에서 노드 추가, 삭제하기

## 노드 삭제하기

노드를 삭제할 때는 부모 노드에서 자식 노드를 삭제해야 한다

□ 노드를 삭제하려면 부모 노드부터 찾아야 함

## parentNode 프로퍼티

현재 노드의 부모 노드에 접근해서 부모 노드의 요소 노드를 반환

기본형 `노드.parentNode`

## removeChild( ) 메서드

자식 노드 삭제

기본형 `부모노드.removeChild(자식노드)`