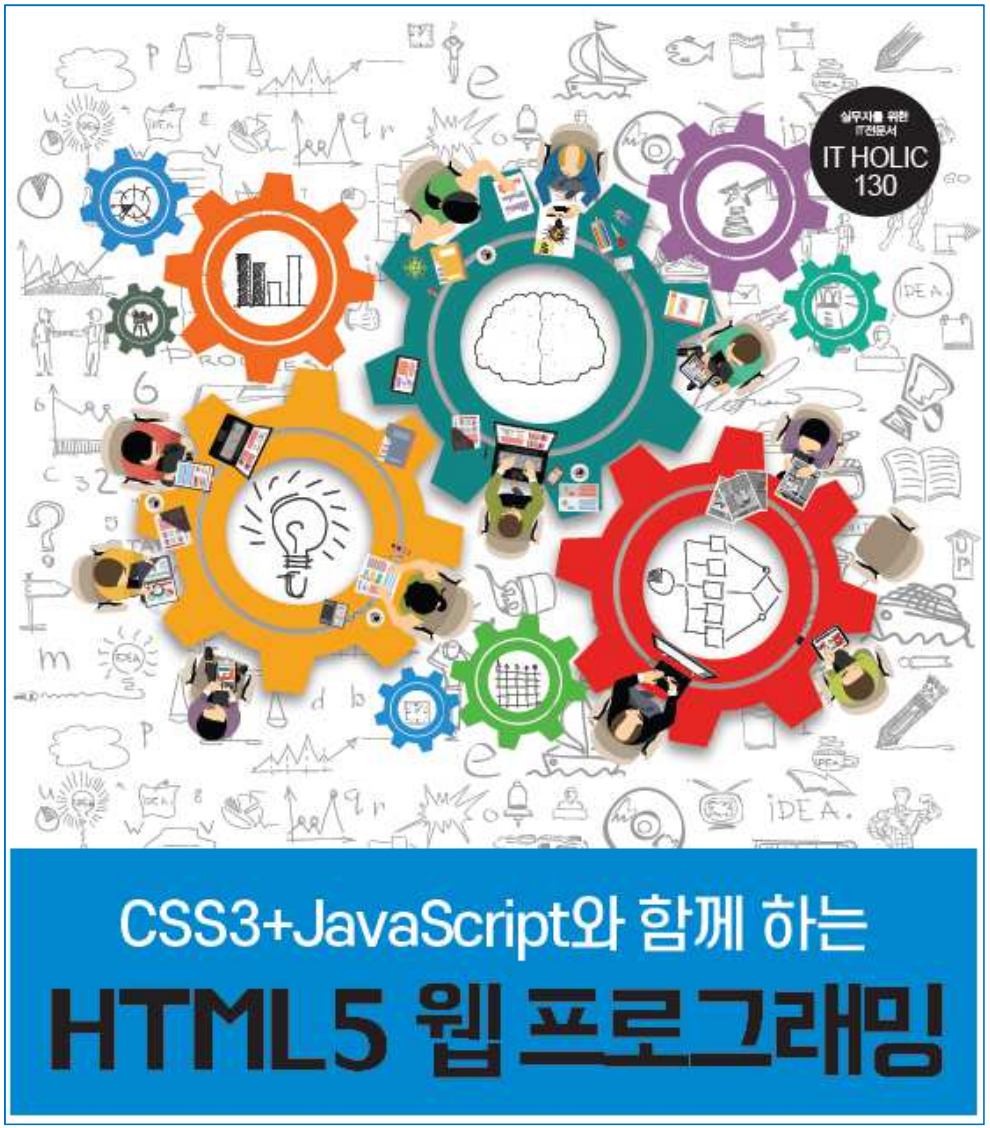



CSS3+JavaScript와 함께 하는

# HTML5 웹 프로그래밍

# 8장 HTML DOM과 동적 문서 작성

CHAPTER

# 08

HTML5+CSS3+JavaScript

# HTML DOM과 동적 문서 작성


8.1 HTML 문서 객체 모델

8.2 DOM을 이용한 HTML 문서 처리

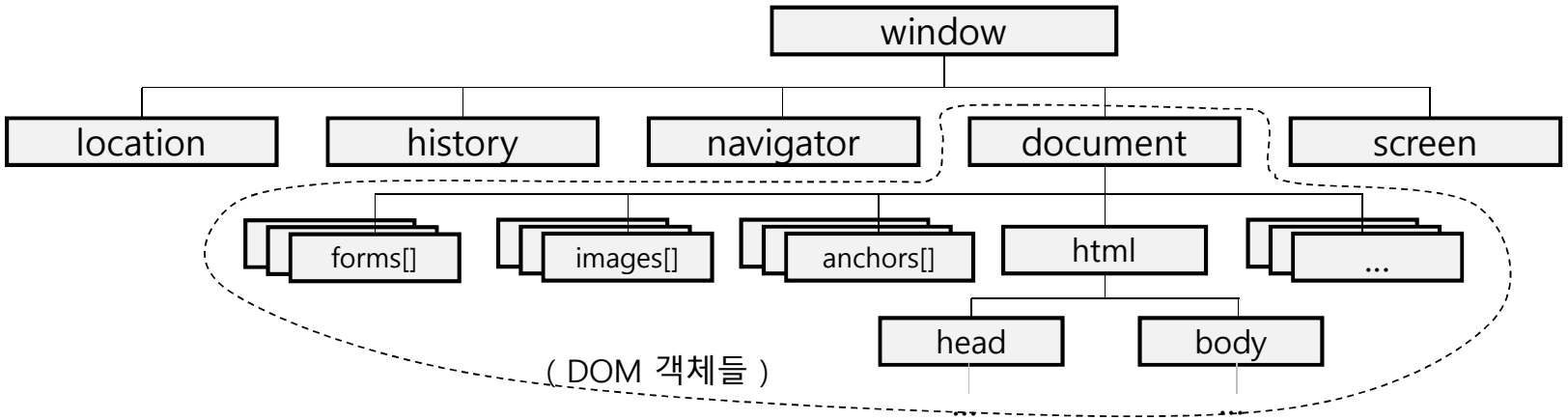
8.3 HTML 입력양식의 데이터 처리

8.4 이벤트 처리

## 8.1 HTML 문서 객체 모델

- DOM(Document Object Model)
  - ✓ W3C에서 규정한 프로그래밍을 이용한 표준 문서접근 방법
  - ✓ 프로그램들이 문서를 접근해서 문서 내용이나 구조, 스타일을 변경할 수 있도록 프로그래밍 인터페이스를 제공함

- HTML DOM
  - ✓ HTML 문서에 대한 DOM 표준
  - ✓ 자바스크립트 코드로 HTML 문서를 접근해서 모든 HTML 요소들을 처리할 수 있는 방법을 제공함
  - ✓ 즉, 웹 브라우저는 HTML 문서를 로드하면, HTML 문서를 document 와 그 하위 객체들로 모델링되어 표현되는 DOM 트리를 생성해서 자바스크립트가 이용할 수 있게 함

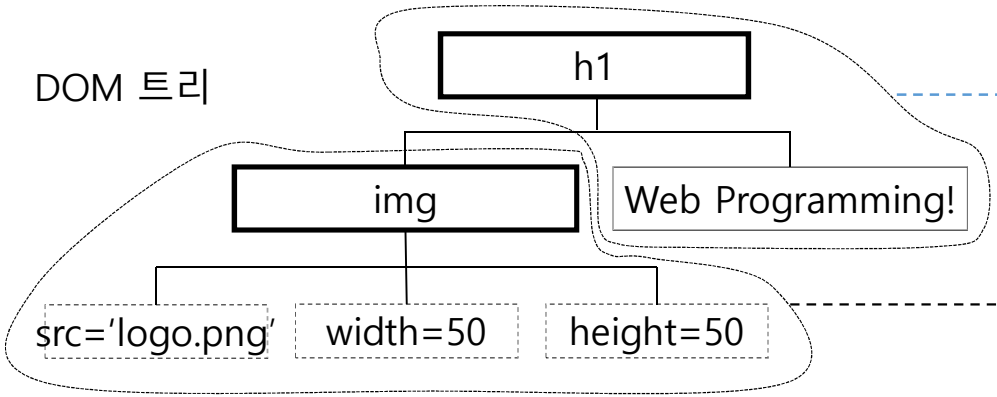


- DOM 트리

  - DOM 표준에 따라 HTML 문서에 있는 HTML 요소들과 그 관계를 객체 및 트리로 나타낸 것
  - 즉, 웹 브라우저는 HTML 문서를 로드하면, DOM 트리를 생성해서 자바스크립트가 이용할 수 있게 함
  - DOM 트리 구조
    - ✓ 루트 : **document** 노드(HTML 문서 자체를 나타냄)
    - ✓ 각 HTML 요소들(태그들) -> **요소노드(element node)**
      - HTML 요소의 텍스트 내용들 -> 텍스트 노드(text node) (항상 DOM 트리의 리프임)
      - -> **요소노드 객체의 innerHTML( 또는 innerText) 속성값이 됨**
      - HTML 요소의 속성들 -> 속성 노드(property node) -> **요소노드 객체의 속성이 됨**

HTML 코드

```
<h1>
   <img src="logo.png" width="50" height="50" />
   Web Programming!
</h1>
```

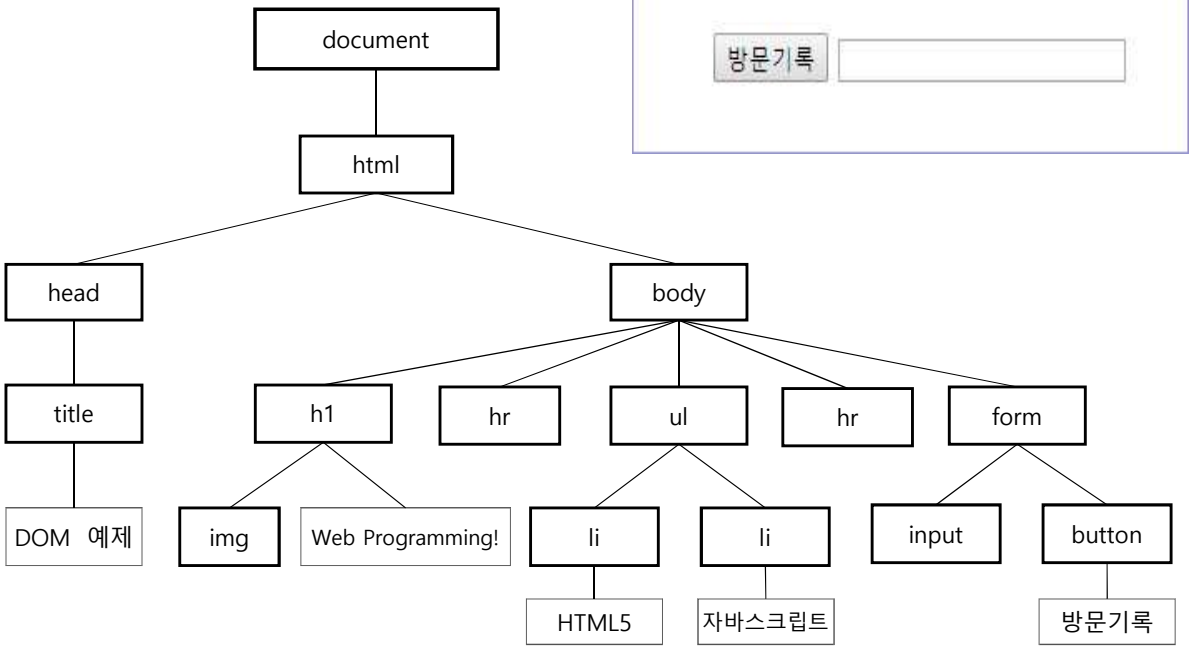



DOM 객체들

```
{ innerHTML: 'Web Programming!'
      ....  }                    (h1 객체)
```

```
{  src : 'logo.png'
   width : 50
   height :  50
    .....  }                     (img 객체)
```

- HTML 문서와 DOM 트리 예


```
<!DOCTYPE html>
<html>
  <head>
    <title> HTML 문서 예 </title>
  </head>
  <body>
    <h2>
    <img src="logo.png"  width="50" height="50" />
      Web Programming !
    </h2>
    <hr>
    <ul>
      <li> HTML5 </li>
      <li> 자바스크립트 </li>
    </u1>
    <br>
    <form>
      <button>방문기록</button>
      <input type="text" />
    </form>
  </body>
</html>
```

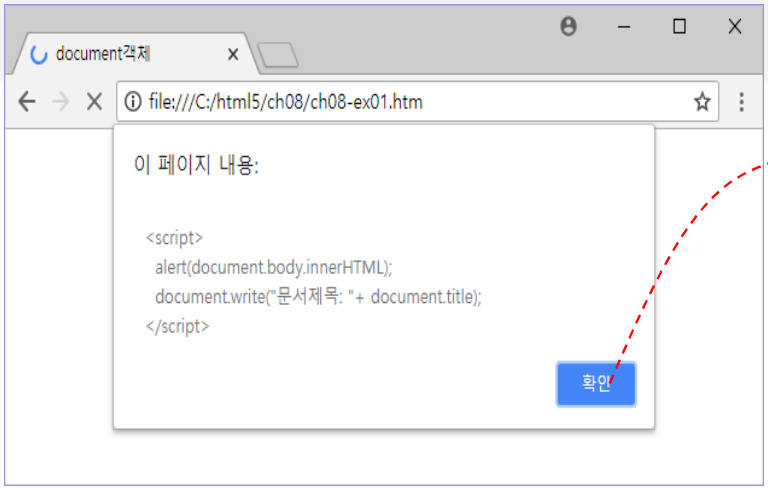
[표 8.1] document의 하위 객체들과 메서드들

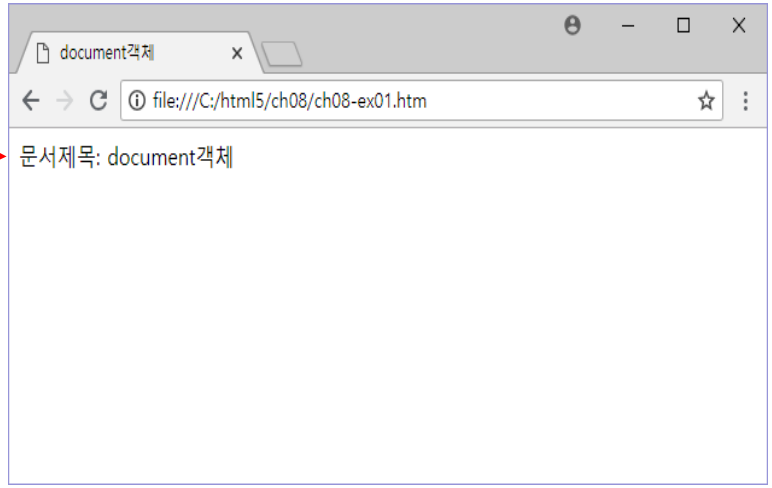
| 하위 객체 및 메서드 | 기능 |
|---|---|
| html | html 요소객체 |
| head | head 요소객체 |
| title | title 요소객체 |
| body | 〈body〉 요소객체 |
| links[ ], scripts[ ], anchors[ ], forms[ ], images[ ] | 각각 모든 〈link〉, 〈script〉, 〈anchor〉, 〈form〉, 〈image〉 요소객체들의 리스트로 구성되는 컬렉션 객체들 |
| getElementById( ) | 요소객체를 반환하는 기본 메서드 |

write()

## [예 8.1] DOM 객체를 이용한 HTML 문서 처리

```
<!DOCTYPE html>
<html>
  <head>
  <title> document객체 </title>
  </head>
<body>
  <script>
    alert(document.body.innerHTML);
    document.write("문서제목: "+ document.title);
  </script>
</body>
</html>
```



## 8.2 DOM을 이용한 HTML 문서 처리

### 8.2.1 DOM을 이용한 HTML 요소 접근

- document 객체(DOM 트리의 루트노드)는 자신의 트리에 있는 특정노드 객체들을 탐색해서 반환하는 여러 메서드들을 제공함
- 대표적인 메서드 : getElementById() – HTML 요소의 id 속성값을 이용해서 탐색함

[HTML 코드]

```
<태그명  id='아이디값'  '.... > 요소내용 </태그명>
```

[자바스크립트 코드]

```
var obj = document.getElementById ('아이디값');     // id='아이디값' 인 요소객체를 탐색함
obj.innerHTML = '변경된 요소내용' ;                  // HTML 요소의 요소내용 변경
obj.attribute = '변경된 속성값' ;                    // HTML 요소의 속성값 변경
obj.style.CSS속성 = '변경된 CSS 속성값' ;            // HTML 요소의 스타일 변경
```

## [그림 8.4] 자바스크립트 DOM 프로그래밍 코드의 실행결과

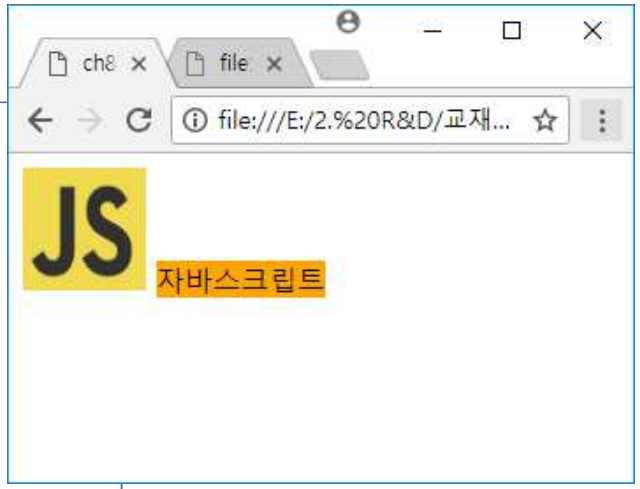
[file1.htm: 변경전]

```
<!DOCTYPE html>
<html>
<head>
  <style>
    #f1 { width:150px; height:150px; }
  </style>
</head>
<body>
  <img id="f1" src="html5.png"/>
  <span id="txt1"> HTML5 </span>
</body>
</html>
```


[dom.js: DOM 프로그래밍 코드]

```
var obj1 = document.getElementById("f1");
obj1.src = "js.png" ;   obj1.style.width="70px";
obj1.style.height = "70px";
var obj2 = document.getElementById("txt1");
obj2.innerHTML = "자바스크립트";
obj2.style.backgroun = "orange";
```

```
<!DOCTYPE html>
<html>
<head>
  <style>
    #f1 { width:70px; height:70px; }
    #txt1 { background : "orange"; }
  </style>
</head>
<body>
  <img id="f1" src="js.png"/>
  <span id="txt1"> 자바스크립트 </span>
</body>
</html>
```




[file2.htm : 변경후]

## 8.2.2 DOM을 이용한 HTML 문서 작성

- DOM 트리 : 현재 웹 브라우저에 로드된 HTML 문서를 객체로 표현한 것임
- 자바스크립트로 DOM 트리를 조작하면 이것은 결과적으로 HTML 문서를 조작하는 것임
- 즉, DOM 트리의 특정 요소노드를 추가, 삭제 또는 그 속성값을 변경하면,
  이것은 HTML 문서에서 대응하는 HTML 요소를 추가, 삭제 또는 그 속성값을 변경하는 것임
  - ✓ 특정 태그 내용의 변경 : 대응하는 요소객체의 innerHTML 속성을 변경함
  - ✓ 특정 태그 속성의 변경 : 대응하는 요소객체의 같은 이름의 속성을 변경함
- **자바스크립트 DOM 프로그래밍 처리 작업**
  - ✓ HTML 요소의 요소내용 참조 및 변경
  - ✓ HTML 요소의 속성값 변경
  - ✓ HTML 요소의 CSS3 스타일 지정
  - ✓ HTML 요소 추가 및 삭제
  - ✓ 애니메이션 효과 구현(HTML 요소의 속성값의 연속적인 변경)

## (1) HTML 태그내용의 속성 변경

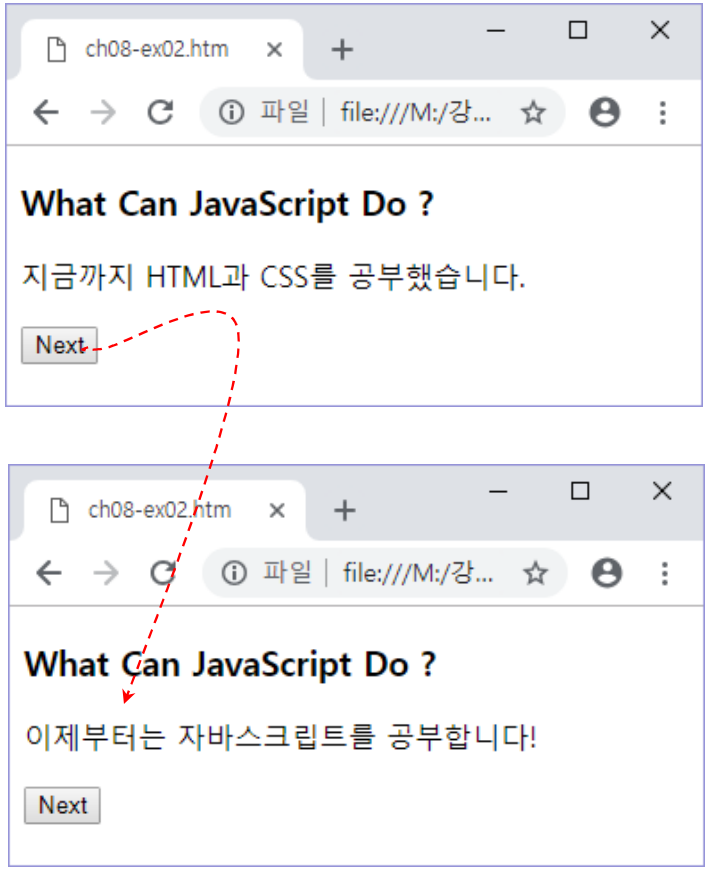
- 특정 HTML 태그내용 변경
  - ✓ 해당 DOM 요소객체의 innerHTML 또는 innerText 속성값을 변경함

- 특정 HTML 요소의 속성 변경
  - ✓ 해당 DOM 요소객체에서 같은 이름 속성의 값을 변경함
  - ✓ 또는 getAttribute() , setAttribute() 메서드 이용

```
var tag1 = document.getElementById("아이디명"); // 속성 id='아이디명'인 요소객체 반환
  ....
// 요소객체의 요소내용, 속성값 변경
tag1.innerHTML = "텍스트(HTML 태그 포함)"; // 또는 tag1.innerText = "텍스트";
tag1.속성명 = "속성값";  // 또는 tag1.setAttribute(속성명, 속성값);

// 요소객체의 요소내용, 속성값 참조
document.write(tag1.innerHTML);  // 또는 document.write(tag1.innerText);
document.write(tag1.속성명); // 또는 document.write(tag1.getAttribute(속성명);
```

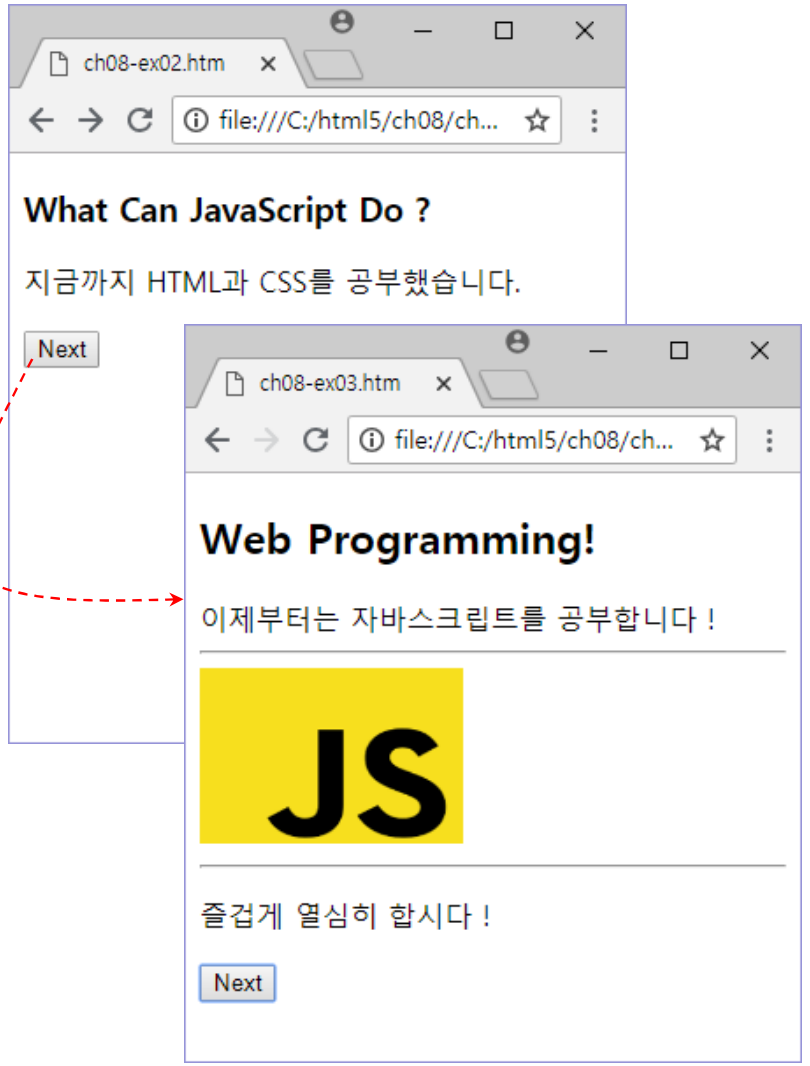
## [예 8.2] 요소내용의 동적인 변경(1) : innerHTML 또는 innerText 속성

```
<script>
    function changeContent() {
        var pTag = document.getElementById("p1");
        pTag.innerHTML = "이제부터는 자바스크립트를 공부합니다!" ;
    }
</script>
<body>
<h3>What Can JavaScript Do ?</h3>
<p id="p1">지금까지 HTML과 CSS를 공부했습니다.</p>
<button type="button"  onclick='changeContent();'> Next </button>
</body>
```



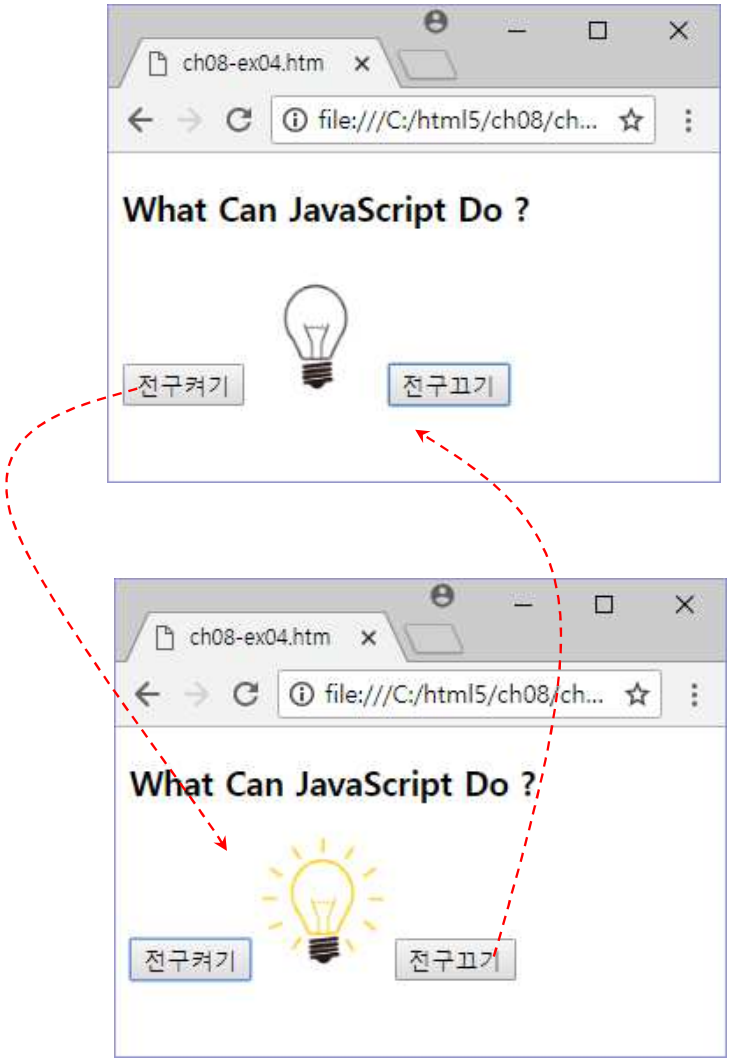
**[예 8.3] 요소내용의 동적인 변경(2) : innerHTML 속성 이용( innerText 속성을 사용하면 안됨)**

```
<script>
    function changeContent() {
        var pTag = document.getElementById("p1");
        var txt = "이제부터는 자바스크립트를 공부합니다 !";
        txt = txt  + "<br><hr>" ;
        txt = txt + "<img src='jslogo.png'  width= 150 height=100 /><hr>";
        txt = txt + "<p> 즐겁게 열심히 합시다 ! </p>";
        pTag.innerHTML = txt ;
    }
</script>
<body>
<h2>Web Programming!</h2>
<p id="p1">지금까지 HTML과 CSS를 공부했습니다.</p>
<button type="button"
   onclick='changeContent();'> Next </button>
</body>
```



## [예 8.4] 요소 속성의 동적인 변경

```
<head>
  <script>
     function changeOff() {
        var tag1 = document.getElementById('light');
        tag1.src='lightoff.png' ;
     }

     function changeOn() {
        var tag1 = document.getElementById('light');
        tag1.src='lighton.png' ;
     }
  </script>
<head>
<body>
  <h3>What Can JavaScript Do ?</h3>
  <button id="b1" onclick="changeOn();">전구켜기</button>
  <img id="light" src="lightoff.png" style="width:70px">
  <button id="b2" onclick="changeOff()">전구끄기</button>
</body>
```



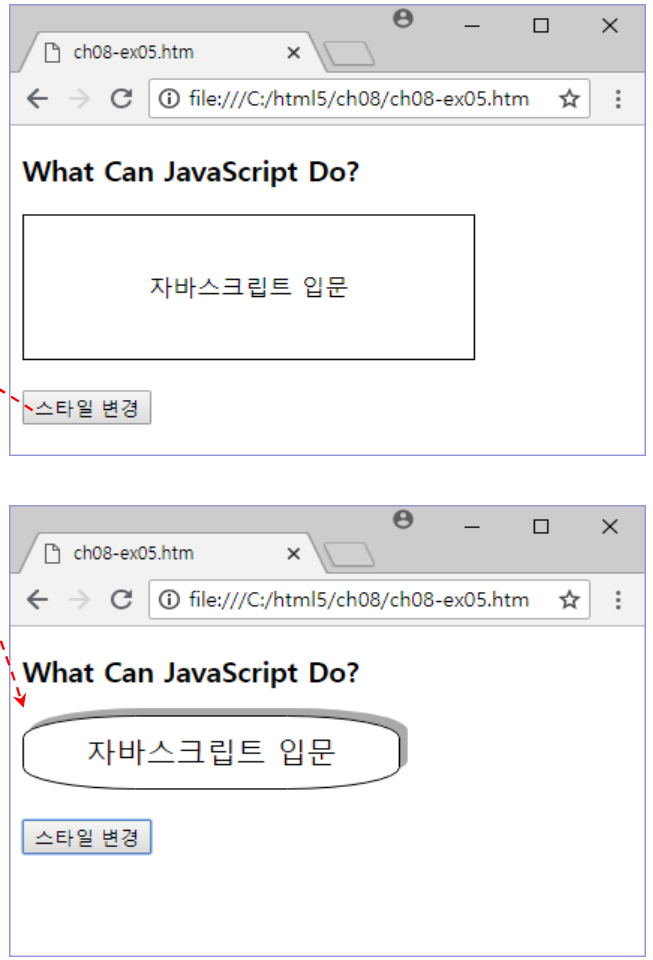
(2) HTML 요소의 스타일 변경

- 해당 요소객체의 하위객체인 **style의 속성 값을 변경**하면 됨
- style 객체의 속성들과 속성값의 표현 방법 : CSS 속성값 지정과 동일함
- 단, 속성이름에서, '–' 대신 '–' 뒤의 단어 첫 글자를 대문자로 해서 붙여야 함
  예) background-color -> **backgroundColor**

```
var tag1 = document.getElementById('ID-명'); // 속성 id='ID-명' 인 요소객체 반환
tag1.style.CSS속성명 = "CSS-속성값";  // 요소객체 tag1.style의 CSS3 스타일 속성값 지정
```

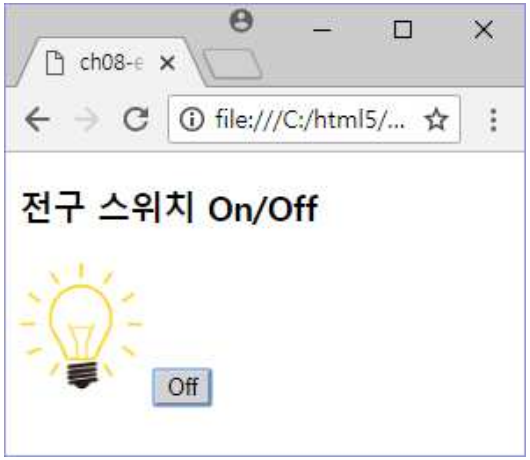
## [예 8.5] 요소 스타일의 동적인 변경

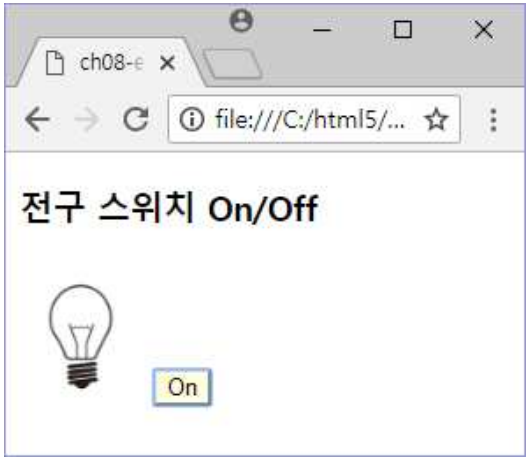
```
<style>
 #b1 {  width : 300px;    height : 100px; text-align : center;
         border : 1px black solid; line-height : 100px;
 }
</style>

<script>
  function changeStyle() {
    var divTag = document.getElementById('b1');
    divTag.style.fontSize='20px';      divTag.style.width = '250px';
    divTag.style.height = '50px';      divTag.style.lineHeight = '50px';
    divTag.style.borderRadius = '30%';
    divTag.style.boxShadow = '5px -5px darkgray';
  }
</script>

<h3>What Can JavaScript Do?</h3>
<div id="b1"> 자바스크립트 입문 </div><br>
<button onclick="changeStyle();"> 스타일 변경</button>
```



[예 8.6] 요소내용, 속성값, 스타일의 동적인 변경

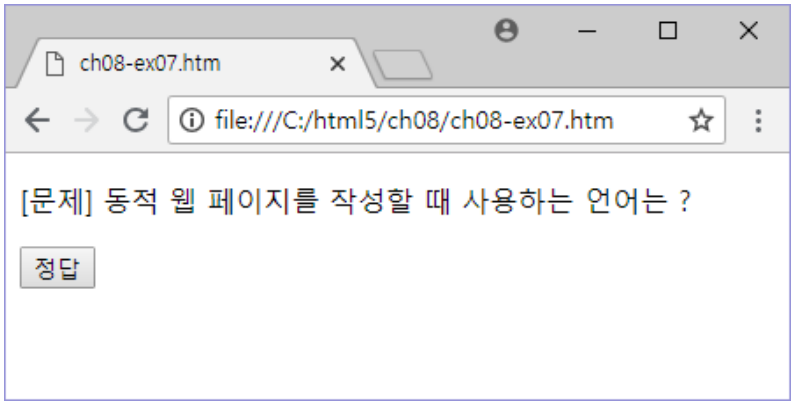
```
<style>
 #switch { background: lightyellow; }
</style>
  ....
<h3> 전구 스위치 On/Off </h3>
<img id="light" src="lightoff.png" style="width:70px">
<button id="switch" onclick="changeSwitch()">On</button>

<script>
  function changeSwitch() {
      var tag1 = document.getElementById('light');
      var tag2 = document.getElementById('switch');
      if (tag2.innerHTML == 'On') {
         tag2.innerHTML = 'Off';
         tag2.style.backgroundColor = "lightgray";
         tag1.src='lighton.png' ;
      }
      else {
         tag2.innerHTML = 'On';
         tag2.style.background = "lightyellow";
         tag1.src='lightoff.png' ;
      }
  }
</script>
```

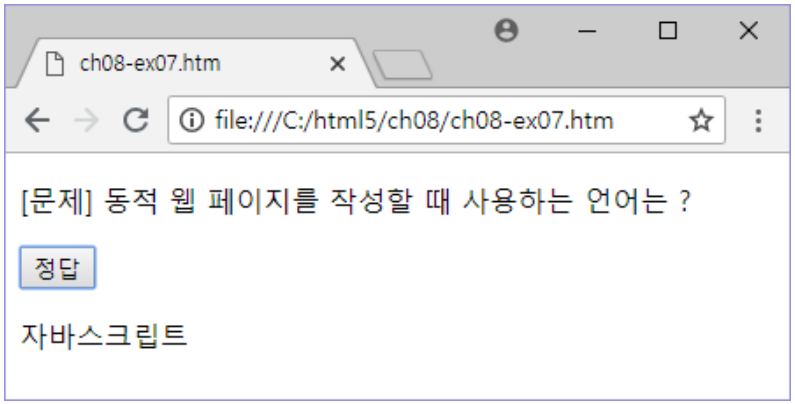


[예 8.7] display 속성을 이용한 동적 출력 표시

```
<p>[문제] 동적 웹 페이지를 작성할 때 사용하는 언어는 ? </p>
<button onclick="showAnswer();">정답</button>
<p id="answer" style="display:none">자바스크립트</p>

<script>
   function showAnswer() {
      var pTag = document.getElementById('answer');
      pTag.style.display = 'block' ;
   }
</script>
```



## (3) HTML 요소의 추가와 삭제, 대체

- DOM 트리에 대응하는 요소 노드를 추가 또는 삭제하면 됨
- 처리방법 : 트리의 노드의 추가, 삭제 연산 방법과 기본적으로 동일함(자료구조)
- document 객체는 DOM 트리의 노드추가, 삭제 작업들을 처리하는 메서드들을 지원함

[표 8.2] DOM 트리 조작 관련 메서드들

| DOM 인터페이스 | 기능 |
|---|---|
| createElement('tagName') | 새로운 요소 노드(<tagName>)를 생성함 |
| createTextNode('textString') | 새로운 텍스트 노드를 생성함 |
| appendChild(childnode) | 자식 노드(childNode)를 추가함 |
| setAttribute(attName, value) | 새로운 속성(attName)과 속성값(value)을 지정함 |
| getAttribute(name) | 속성값을 참조함 |
| removeChild(child) | 자식 노드(childNode)를 제거함 |
| insertBefore(newC, node) | 특정 자식 노드(node)의 직전 형제 노드로 추가함 |
| replaceChild(new, old) | 요소 노드(old)를 새로운 요소 노드(new)로 대체함 |

## 3-1. DOM 트리의 요소 노드 추가

① 먼저 DOM 트리에 추가할 요소 노드(newTag)와 텍스트 노드(tagContent)를 생성 한 후, 생성한 요소 노드(newTag)에 텍스트 노드 (tagContent)를 자식노드로 연결함

```
var newTag = document.createElement('태그명'); // 요소 노드 생성
var tagContent = document.createTextNode('태그내용'); // 텍스트 노드 생성
newTag.appendChild(tagContent); // 텍스트 노드를 요소 노드의 자식 노드로 연결함
```

② 다음으로, 생성한 요소노드(newTag)의 필요한 속성 노드들을 추가함

```
// 요소 노드 newTag에 필요한 속성을 지정함
newTag.setAttribute('속성명1', '속성값1');
newTag.setAttribute('속성명2', '속성값2');
```

특정 요소노드(siblingTag)의 직전 형제노드로 추가하려면, 그 부모노드의 insertBefore() 메서드를 이용함

## 3-1. DOM 트리의 요소 노드 추가(계속)

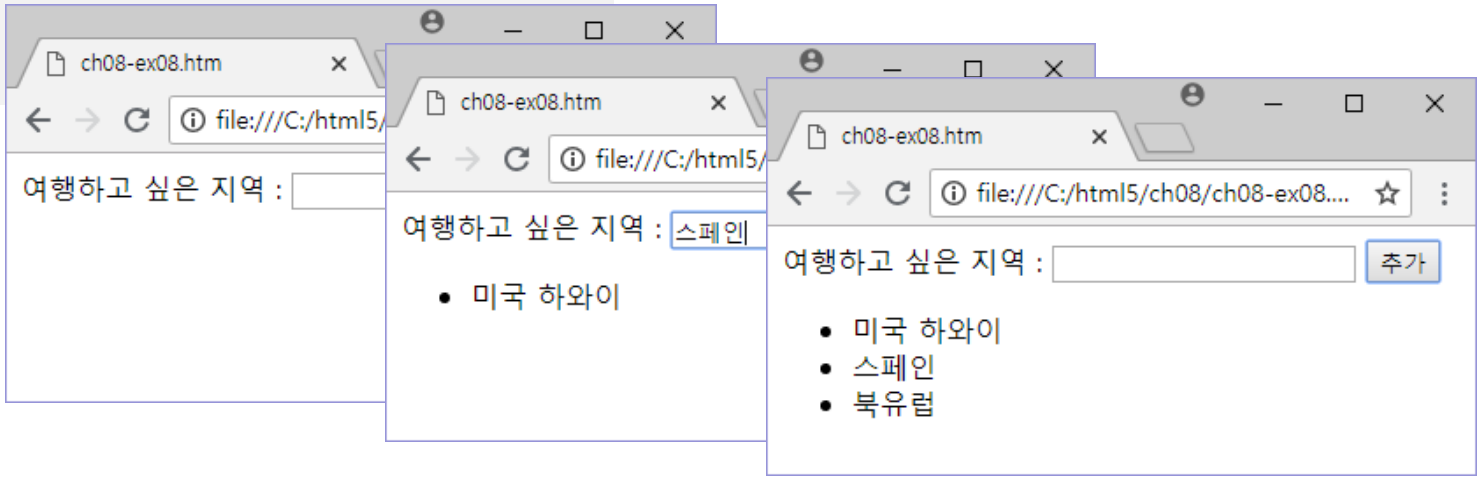
③ 마지막으로, DOM 트리에서 생성한 요소 노드(newTag)의 부모 요소 노드를 탐색하여, 생성한 요소 노드(newTag)를 그 자식노드로 연결함(마지막 자식노드로 추가됨).

```
// 부모 노드(parentTag)의 마지막 자식 노드로 연결함
var parentTag = document.getElementById('아이디명'); // 부모 요소 노드 탐색
parentTag.appendChild(newTag); // newTag를 parentTag의 자식 노드로 추가함

// 특정 노드(siblingTag)의 직전 형제 노드로 추가함
var parentTag = document.getElementById('p1'); // 부모 요소 노드(p1) 탐색
var siblingTag = document.getElementById('s1'); // 형제 요소 노드(s1) 탐색
parentTag.insertBefore(newTag, siblingTag); // newTag를 sibling의 직전 형제 노드로 추가함
```

✓ 특정 요소노드(siblingTag)의 직전 형제노드로 추가하려면,
그 부모노드의 insertBefore() 메서드를 이용함

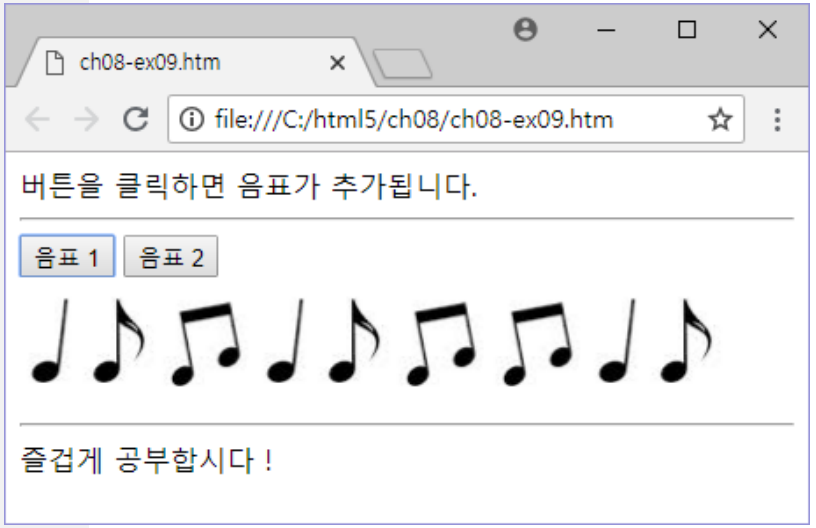
**[예 8.8] 사용자 입력의 문서 내용 추가**

```
여행하고 싶은 지역 : <input id="t1" type="text" />
 <button onclick="listPlace();" >추가</button>
 <ul id="t2" ></ul>
 <script>
   function listPlace() {
     var inTag = document.getElementById('t1');
     var listPlace = document.getElementById('t2');
     var  newPlace = document.createElement('li');
     var  placeNode = document.createTextNode(inTag.value);
     newPlace.appendChild(placeNode);
     listPlace.appendChild(newPlace);
     inTag.value='';
   }
 </script>
```



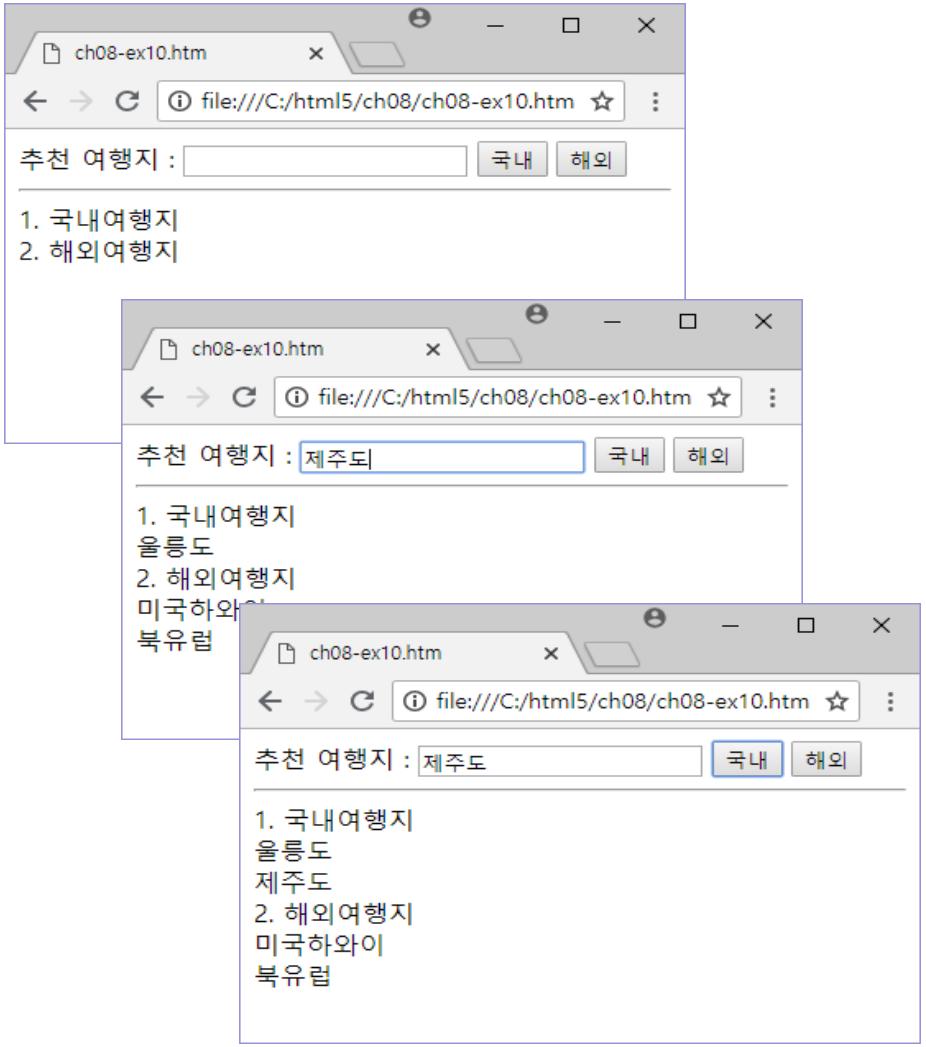
[예 8.9] 문서 내용으로 이미지 추가

```
버튼을 클릭하면 음표가 추가됩니다. <hr>
 <button onclick="insertImage1();" >음표 1</button>
 <button onclick="insertImage2();" >음표 2</button>

 <div id="b1"> </div>
 <hr> 즐겁게 공부합시다 !
 <script>
   function insertImage1() {
     var tag1 = document.getElementById('b1');
     var newImage = document.createElement('img');
     newImage.setAttribute('src', 'm1.png');
     tag1.appendChild(newImage);
   }

   function insertImage2() {
     var tag1 = document.getElementById('b1');
     var  newImage = document.createElement('img');
     newImage.setAttribute('src', 'm2.png');
     tag1.appendChild(newImage);
   }
 </script>
```



## [예 8.10] 특정 노드의 직전 형제 노드 추가

```
추천 여행지 :
  <input id="t1" type="text" />
  <button onclick="listPlace1();" >국내</button>
  <button onclick="listPlace2();" >해외</button>  <hr>
  <div id="d1">
    <div>1. 국내여행지</div>
    <div id="d2">2. 해외여행지</div>
 </div>
 <script>
    function listPlace2() {
      var tag1 = document.getElementById('t1');
      var tag2 = document.getElementById('d1');
      var newtag = document.createElement('div');
      var txtNode = document.createTextNode(tag1.value);
      newtag.appendChild(txtNode);
      tag2.appendChild(newtag);  inTag.value='';
    }
    function listPlace1() {
      var tag1 = document.getElementById('t1');
      var tag2 = document.getElementById('d1');
      var tag3 = document.getElementById('d2');
      var newtag = document.createElement('div');
      var txtNode = document.createTextNode(tag1.value);
      newtag.appendChild(txtNode);
      tag2.insertBefore(newtag, tag3);  inTag.value='';
    }
 </script>
```



## 3-2. DOM 트리의 요소 노드 삭제와 대체

- 특정 요소노드의 삭제 처리
  - ✓ 그 부모 요소노드를 탐색해서 **부모노드의 removeChild() 메서드를 이용해서 삭제함**
  - ✓ 각 요소노드의 **부모노드는 parentNode 속성 객체로 제공됨**

```
var deletedTag = document.getElementById('p');    // 삭제할 요소 노드 탐색
deletedTag.parentNode.removeChild(deletedTag);  // 요소 노드(p)를 삭제함
```

- 특정 요소노드(node1)를 새로운 요소노드(node2)로 대체 처리
  1) 먼저 새로운 요소 노드(node2)를 생성하고
  2) 대체할 노드(node2)의 부모노드를 탐색해서 **부모노드의 replaceChild() 메서드를 이용**해서 다음과 같이 대체함

```
// 신규 요소 노드 생성
var newTag = document.createElement('태그명');   // 요소 노드 생성
var tagContent = document.createTextNode('태그내용.');  // 텍스트 노드 생성
newTag.appendChild(tagContent); // 텍스트 노드를 요소 노드의 자식 노드로 연결함

var oldTag = document.getElementById('p'); // 대체할 요소 노드(oldTag) 탐색
var parent = oldTag.parentNode; // oldTag의 부모 요소 노드(p) 탐색
parent.replaceChild(newTag, oldTag);
```

```
<h3> 웹브라우저 <h3>
<div id="p1">
   <img src="wb1.png" width=60 height=60 />
   <img src="wb2.png" width=60 height=60 id="w1" />
   <img src="wb3.png" width=60 height=60 />
   <img src="wb4.png" width=60 height=60 id="w2" />
   <img src="wb5.png" width=60 height=60 />
   <img src="wb6.png" width=60 height=60 />
</div><hr>
<button onclick="deleteNetscape();" >넷스케이프 삭제</button>
<button onclick="replaceSapari();" >사파리 대체</button> <hr>

<script>
   function deleteNetscape(){
      var  deletedTag = document.getElementById('w1');
      deletedTag.parentNode.removeChild(deletedTag);
   }

   function replaceSapari() {
      var  newTag = document.createElement('span');
      var  tagContent = document.createTextNode('사파리');
      newTag.appendChild(tagContent);
      var  oldTag = document.getElementById('w2');
      var  parent = oldTag.parentNode;
      parent.replaceChild(newTag, oldTag);
   }
</script>
```

## [예 8.11] 이미지 요소의 삭제와 대체

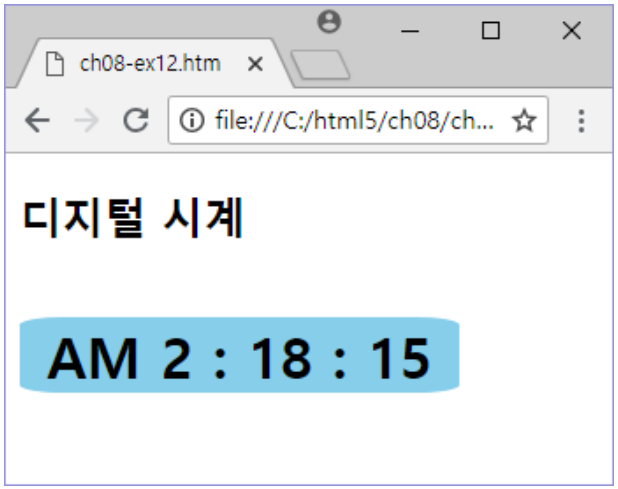
## (4) 애니메이션 효과 구현

- 애니메이션 효과 : HTML 요소의 태그내용이나 속성값을 짧은 시간간격으로 계속 변경함
- 이러한 변경의 주기적인 반복 : window.setInterval() 메서드 이용
- **window.setInterval()** 사용 방법 : 7.2절 참고

```
var interTime = 1000    // 애니메이션 효과 속성값의 변경 시간 간격(밀리초) 지정
function changeObj() { // 애니메이션 효과 관련 속성값 변경 함수 정의
    var obj = document.getElementById(); // 애니메이션 효과를 나타낼 요소객체 접근
                    ....
    obj.attr1 = valueFunction();
}
setInterval( changeObj, interTime);
```

**[예 8.12] 디지털 시계 구현**

```
<style>
 #t1 {
   width: 250px;              background : skyblue;
   text-align : center;       border-radius : 10%;
   display : inline-block;
 }
</style>

<script>
  var inter = 500 ;
  function changeObj() {
    var obj = document.getElementById('t1');
    var d = new Date();   var h = d.getHours();
    var m = d.getMinutes();   var s = d.getSeconds();
    if ( h < 12)  txt = "AM ";
    else  txt = "PM ";
    obj.innerHTML =  txt + "  "+ h + " : " + m + " : " + s + "  " ;
  }
  setInterval(changeObj, inter);
</script>
    ....
<h2>디지털 시계</h2>
<h1 id="t1">  </h1>
```



```
<style>
 #rangeProgress {   width: 100%;        height: 30px;
    position: relative;      background-color: darkgray;
 }

 #bar {    background-color: orange;
     width: 0px;   height: 30px;    position: absolute;
 }
</style>
  ...
<h3> 진행과정 표시(Progress Bar) </h3>
  진행상태 : <span id="demo"> 0%</span>  <hr>
<div id="rangeProgress">    <div id="bar"></div> </div><hr>
<button onclick="move()">진행</button>

<script>
  function move() {
      var progressBar = document.getElementById("bar");
      var demo1 = document.getElementById("demo");
      var width = 0;
      var id = setInterval(function frame() {
        if (width == 100) clearInterval(id);
        else {
          width++;      progressBar.style.width = width + '%';
          demo1.innerHTML= width + '%';
        }
      }, 500);
    }
  </script>
```

## [예 8.13] 진행상태 표시(Progress Bar)