09

이벤트 기초 및 활용

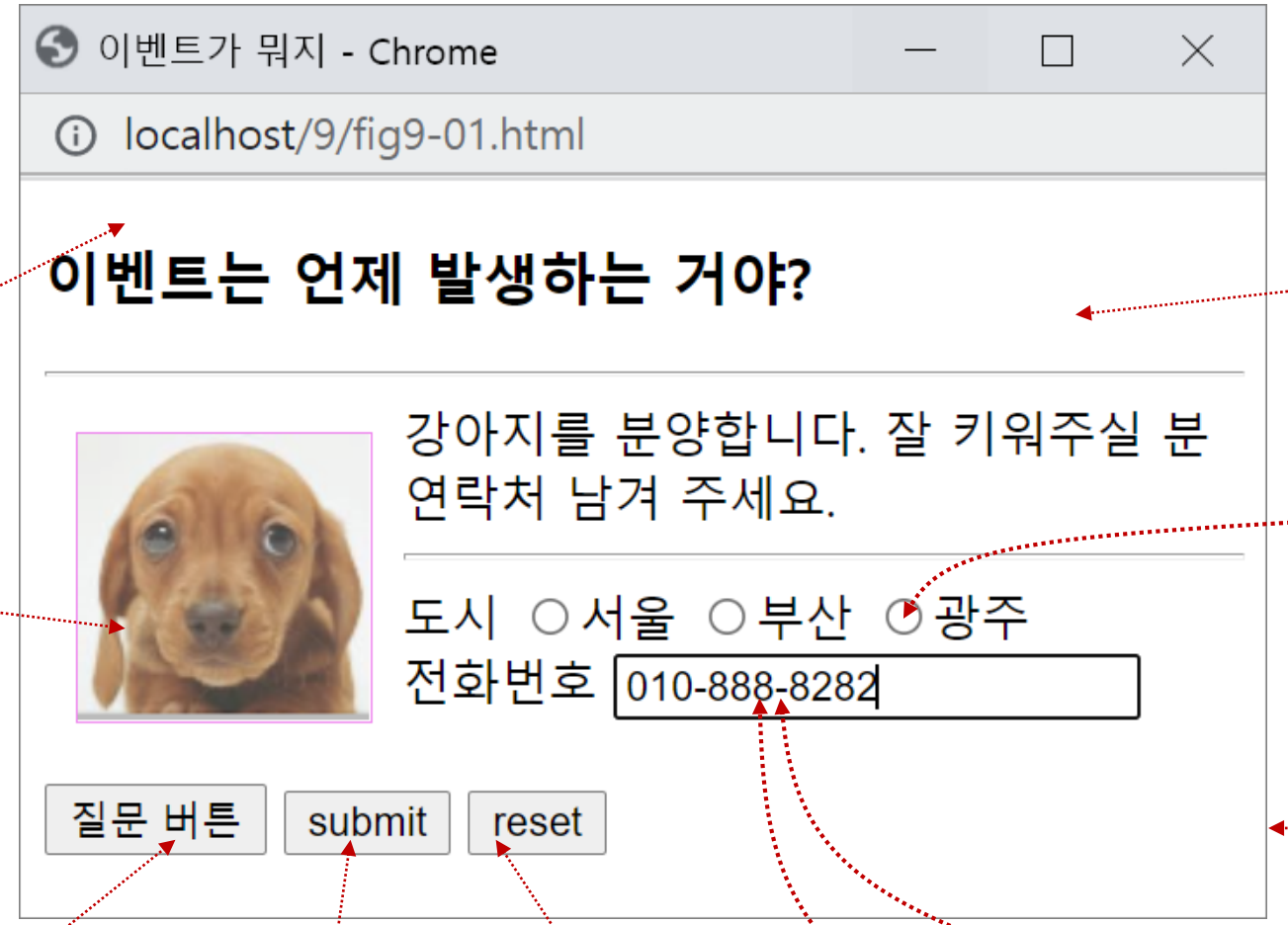
강의 목표

1. 이벤트가 무엇이고 언제 발생하는지 안다.
2. 자바스크립트 코드로 이벤트 리스너를 작성할 수 있다.
3. 발생하는 이벤트가 DOM 트리를 따라 흘러가는 경로를 안다.
4. 문서와 이미지의 로딩 완료 시 호출되는 onload 리스너를 작성할 수 있다.
5. 폼에 발생하는 이벤트 리스너를 다룰 수 있다.
6. 마우스 관련 이벤트를 다룰 수 있다.
7. 키 관련 이벤트를 다룰 수 있다.

이벤트 개요

- 이벤트
  - 마우스 클릭, 키보드 입력, 이미지나 HTML 문서의 로딩, 타이머의 타임아웃 등 사용자의 입력 행위나 문서나 브라우저의 상태 변화를 자바스크립트 코드에게 알리는 통지(notification)
- 이벤트 리스너
  - 발생한 이벤트에 대처하기 위해 작성된 자바스크립트 코드
- 이벤트 종류
  - HTML5에서 이벤트 종류는 70여가지
  - 이벤트 리스너 이름은 이벤트 이름 앞에 on을 덧붙임
  - 예) onmousedown : mousedown 이벤트의 리스너
    onkeydown : keydown 이벤트의 리스너

브라우저에 발생하는 다양한 이벤트들



load 이벤트 (HTML 문서 전체 로딩 완료 시)

dblclick 이벤트 (마우스 더블클릭 시)

load 이벤트 (이미지의 로딩 완료 시)

change 이벤트 (라디오버튼 선택 시)

resize 이벤트 (윈도우 크기 변경 시)

click 이벤트 (마우스 클릭 시)

submit 이벤트 (submit 버튼 클릭 시)

reset 이벤트 (reset 버튼 클릭 시)

keydown 이벤트 (키를 누를 때)

keyup 이벤트 (누른 키를 놓을 때)

이벤트 리스너 만들기

- 3 가지 방법
  - HTML 태그 내에 작성
  - DOM 객체의 이벤트 리스너 프로퍼티에 작성
  - DOM 객체의 addEventListener() 메소드 이용

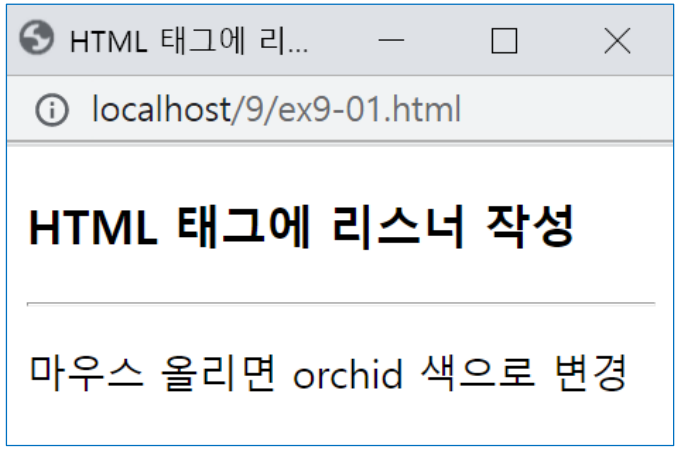
- HTML 태그 내에 이벤트 리스너 작성
  - HTML 태그의 이벤트 리스너 속성에 리스너 코드 직접 작성

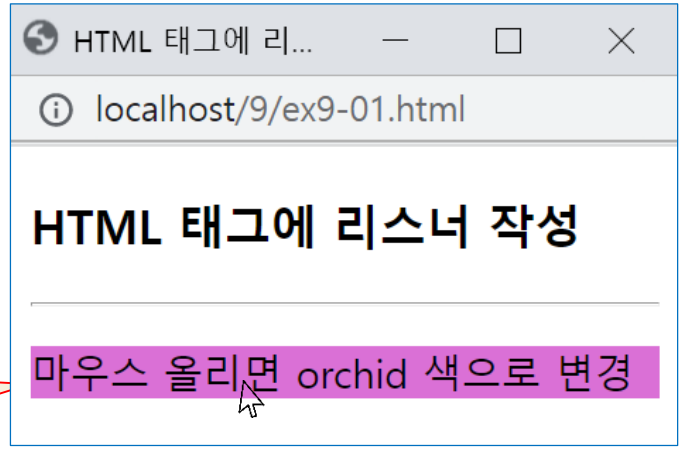
    예) <p>태그에 마우스 올리면 orchid, 내리면 흰색으로 배경변경

```
<p onmouseover="this.style.backgroundColor='orchid'"
    onmouseout="this.style.backgroundColor='white'">
       마우스 올리면 orchid 색으로 변경
</p>
```

예제 9-1 HTML 태그 내에 이벤트 리스너 작성


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>HTML 태그에 리스너 작성</title>
</head>
<body>
<p>HTML 태그에 리스너 작성</p>
<hr>
<p onmouseover="this.style.backgroundColor='orchid'"
   onmouseout="this.style.backgroundColor='white'">
  마우스 올리면 orchid 색으로 변경</p>
</body>
</html>
```





이곳에 마우스를 올리면 배경색 변함

DOM 객체의 이벤트 리스너 프로퍼티에 작성

- DOM 객체의 이벤트 리스너 프로퍼티에 이벤트 리스너 코드 작성

예)

```
<p id="p">마우스 올리면 orchid 색으로 변경</p>
```

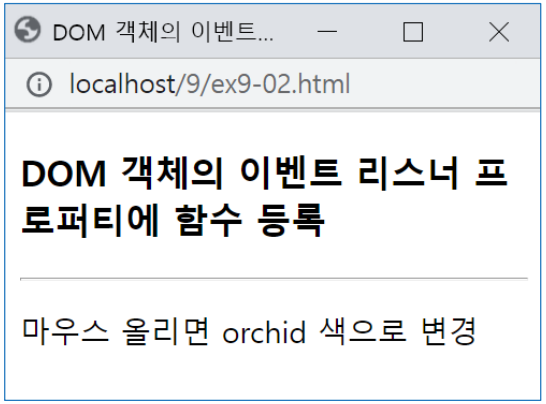
```
function over() { // onmouseover 리스너로 사용할 함수
   ...
}
```

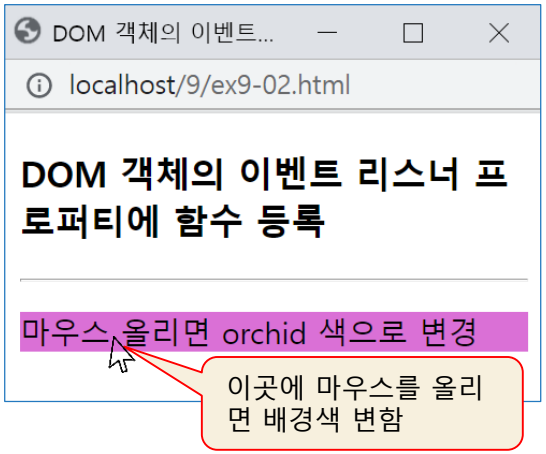
```
let p = document.getElementById("p");
p.onmouseover = over;      // onmouseover 리스너로 over() 함수 등록
```

```
p.onmouseover = over(); // 잘못된 코드
```

예제 9-2 DOM 객체의 이벤트 리스너 프로퍼티에 리스너 등록


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>DOM 객체의 이벤트 리스너 프로퍼티에 함수 등록</title>
<script>
let p;
function init() { // 문서가 완전히 로드되었을 때 호출
    p = document.getElementById("p");
    p.onmouseover = over; // over()를 onmouseover 리스너로 등록
    p.onmouseout = out; // out()를 onmouseout 리스너로 등록
}
function over() {
    p.style.backgroundColor="orchid";
}
function out() {
    p.style.backgroundColor="white";
}
</script>
</head>
<body onload="init()">
<h3>DOM 객체의 이벤트 리스너 프로퍼티에 함수 등록</h3>
<hr>
<p id="p">마우스 올리면 orchid 색으로 변경</p>
</body>
</html>
```





DOM 객체의 addEventListener() 메소드 활용

- addEventListener() 메소드

addEventListener(*eventName*, *listener*[, *useCapture*])

- eventName : 이벤트 타입을 나타내는 문자열. click, load, keydown 등
- listener : 이벤트 리스너로 등록할 함수 이름
- useCapture : true이면 이벤트 흐름 중 캡쳐 단계에서 실행될 리스너(listener 함수) 등록.
  false이면 버블 단계에서 실행될 리스너 등록. 생략 가능하며 디폴트는 false.
  이벤트 흐름은 3절에서 자세히 설명

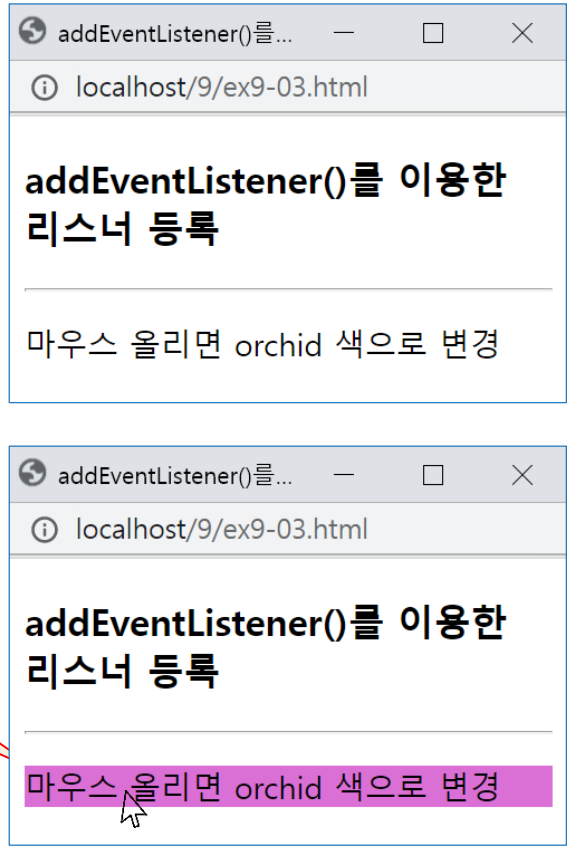
listener 함수를 eventName의 이벤트를 처리할 리스너로 등록한다.

예)

```
p.addEventListener("mouseover", over); // onmouseover 리스너로 over() 등록
```

예제 9–3 addEventListener() 사용


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>addEventListener()를 이용한 리스너 등록</title>
<script>
let p;
function init() { // 문서가 완전히 로드되었을 때 호출
    p = document.getElementById("p");
    p.addEventListener("mouseover", over); // 이벤트 리스너 등록
    p.addEventListener("mouseout", out); // 이벤트  리스너 등록
}
function over() {
    p.style.backgroundColor="orchid";
}
function out() {
    p.style.backgroundColor="white";
}
</script>
</head>
<body onload="init()">
<h3>addEventListener()를 이용한 리스너 등록</h3>
<hr>
<p id="p">마우스 올리면 orchid 색으로 변경</p>
</body>
</html>
```



익명 함수로 이벤트 리스너 작성

- 익명 함수(anonymous function)
  - 함수 이름 없이 필요한 곳에 함수의 코드를 바로 작성

    예)

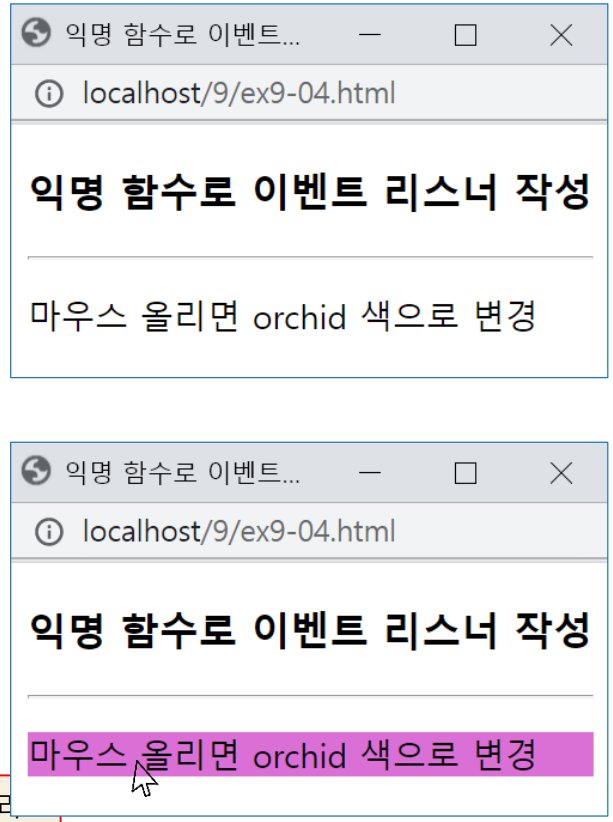
```
p.onmouseover = function () { this.style.backgroundColor = "orchid"; }; // 익명 함수
```

```
p.addEventListener("mouseover",
      function () { this.style.backgroundColor = "orchid"; } // 익명 함수
);
```

  - 코드가 짧거나 한 곳에서만 사용하는 경우, 익명 함수 편리

예제 9-4 익명 함수로 이벤트 리스너 작성


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>익명 함수로 이벤트 리스너 작성</title>
<script>
let p;
function init() { // 문서가 완전히 로드되었을 때 호출
    p = document.getElementById("p");
    p.onmouseover = function () { // 익명 함수
        this.style.backgroundColor = "orchid";
    };
    p.addEventListener("mouseout",
        function () { this.style.backgroundColor="white"; } // 익명 함수
    );
}
</script>
</head>
<body onload="init()">
<h3>익명 함수로 이벤트 리스너 작성</h3>
<hr>
<p id="p">마우스 올리면 orchid 색으로 변경</p>
</body>
</html>
```



이곳에 마우스를 올리면 배경색 변함

이벤트 리스너 작성 방법 4 가지 비교


```
function over() {
    p.style.backgroundColor="orchid";
}
```

(1) HTML 태그

```
<p id="p" onmouseover="this.style.backgroundColor='orchid'"
   마우스 올리면 orchid 색으로 변경
</p>
```

(2) 이벤트 리스너 프로퍼티

```
function over() {
   p.style.backgroundColor="orchid";
}
p.onmouseover = over;
```

(3) addEventListener() 메소드 이용

```
p.addEventListener("mouseover", over);
```

(4) 익명 함수 이용

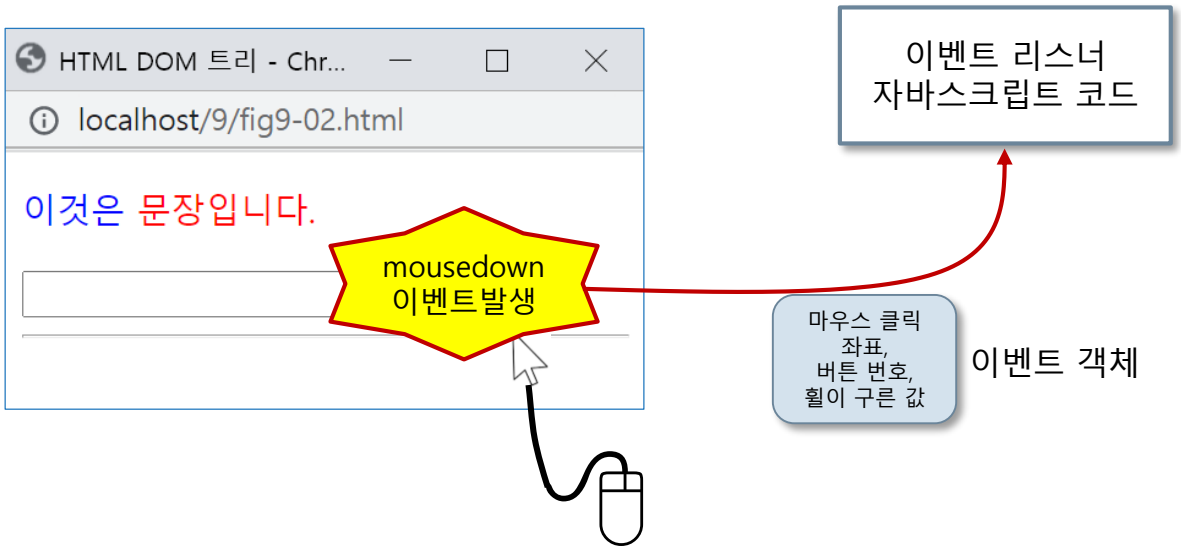
```
p.onmouseover = function () { this.style.backgroundColor="orchid"; };
```

(5) 익명 함수 이용

```
p.addEventListener("mouseover",
   function () { this.style.backgroundColor="orchid"; }
);
```

이벤트 객체

- 이벤트 객체(event object)
  - 발생한 이벤트에 관련된 다양한 정보를 담은 객체
  - 예) mousedown 이벤트의 경우, 마우스 좌표와 버튼 번호 등
    keydown 이벤트의 경우, 키 코드 값 등
  - 이벤트가 처리되고 나면 이벤트 객체 소멸



이벤트 객체 전달받기

- 이벤트 객체는 이벤트 리스너 함수의 첫 번째 매개변수에 전달

1. 이름을 가진 이벤트 리스너

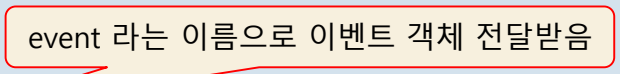
```
function f(e) { // 매개변수 e에 이벤트 객체 전달받음
        ...
}
obj.onclick = f; // obj 객체의 onclick 리스너로 함수 f 등록
```

2. 익명 함수의 경우

```
obj.onclick = function(e) { // 매개변수 e에 이벤트 객체 전달받음
        ...
}
```

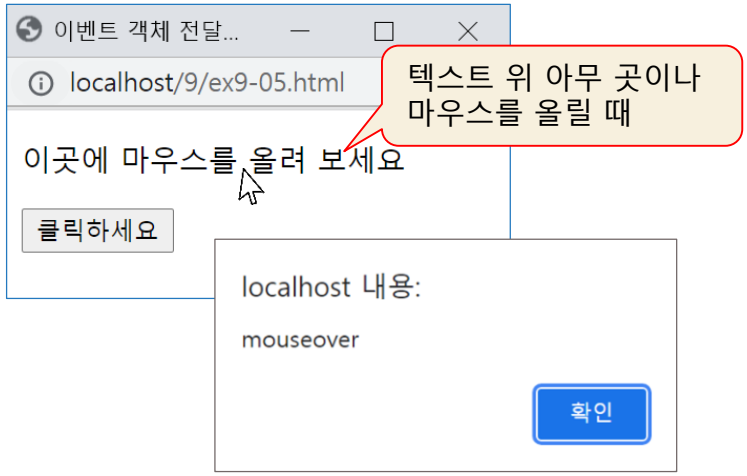
3. HTML 태그에 이벤트 리스너 : **event** 라는 이름으로 전달

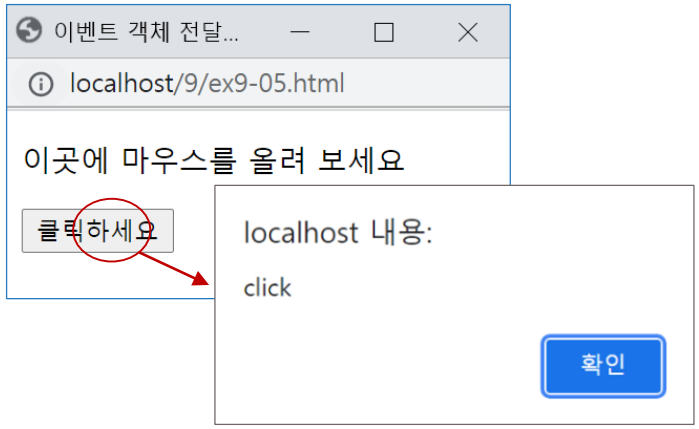
```
function f(e) {
        ...
}
        ...
<button onclick="f(event)">버튼</button>
<div onclick="alert(event.type)">버튼</div>
```



예제 9-5 이벤트 리스너에서 이벤트 객체 전달 받기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>이벤트 객체 전달받기</title>
</head>
<body>
<p id="p">이곳에 마우스를 올려 보세요</p>
<button onclick="f(event)">클릭하세요</button>
<script>
function f(e) { // e는 현재 발생한 이벤트 객체
   alert(e.type); // 이벤트 종류 출력
}

document.getElementById("p").onmouseover = f;
</script>
</body>
</html>
```





이벤트 객체에 들어 있는 정보

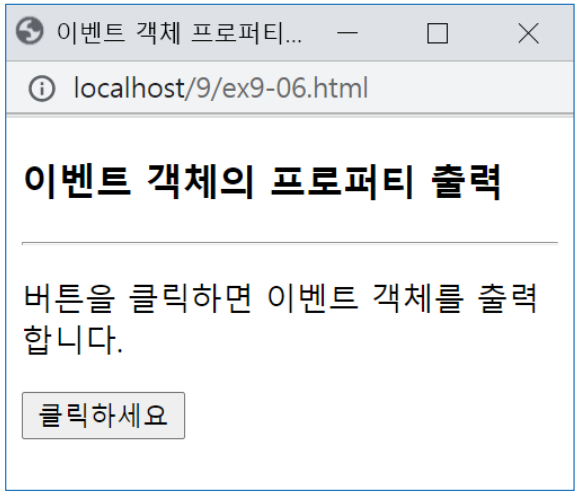
- 이벤트 객체에 들어 있는 정보
  - 현재 발생한 이벤트에 관한 다양한 정보
    - 이벤트 객체의 프로퍼티와 메소드로 알 수 있음
  - 이벤트의 종류마다 조금씩 다름
    - 이벤트 객체의 공통 멤버

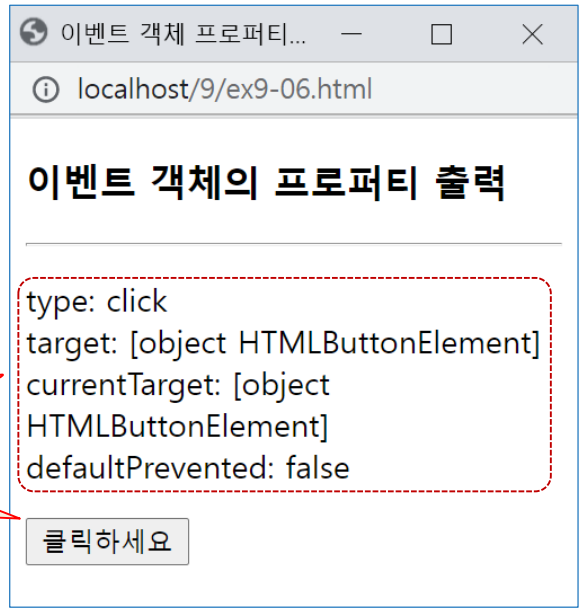
멤버	종류	설명
type	프로퍼티	현재 발생한 이벤트의 종류를 나타내는 문자열(click, load 등)
target	프로퍼티	이벤트를 발생시킨 객체(DOM 객체 혹은 HTML 태그)
currentTarget	프로퍼티	현재 이벤트 리스너를 실행하고 있는 DOM 객체
defaultPrevented	프로퍼티	이벤트의 디폴트 행동이 취소되었는지를 나타내는 true/false 값
preventDefault()	메소드	이벤트의 디폴트 행동을 취소시키는 메소드

  - target 프로퍼티
    - 이벤트 타겟 객체 가리킴
    - 이벤트 타겟 : 이벤트를 유발시킨 DOM 객체
      - *<button> 태그의 버튼을 클릭하였으면, 이때 click 이벤트의 이벤트 타겟은 버튼*

예제 9-6 이벤트 객체의 프로퍼티 출력


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>이벤트 객체 프로퍼티</title>
</head>
<body>
<h3>이벤트 객체의 프로퍼티 출력</h3>
<hr>
<p id="p">버튼을 클릭하면 이벤트 객체를 출력합니다.</p>
<button onclick="f(event)">클릭하세요</button>
<script>
function f(e) { // e는 현재 발생한 이벤트 객체
   let text = "type: " + e.type + "<br>"
              + "target: " + e.target + "<br>"
              + "currentTarget: " + e.currentTarget + "<br>"
              + "defaultPrevented: " + e.defaultPrevented;

   let p = document.getElementById("p");
   p.innerHTML = text; // 이벤트 객체의 프로퍼티 출력
}
</script>
</body>
</html>
```





버튼을 클릭하면 click 이벤트 객체의 프로퍼티 출력

이벤트의 디폴트 행동 취소, preventDefault()

- 이벤트의 디폴트 행동이란?
  - 특정 이벤트에 대한 HTML 태그의 기본 행동
  - 사례
    - <a>의 click 이벤트의 디폴트 행동 : 웹 페이지 이동
    - submit 버튼의 click 이벤트의 디폴트 행동 : 폼 데이터 전송
    - <input type="checkbox">의 click 이벤트의 디폴트 행동 : 체크박스선택
- 이벤트의 디폴트 행동을 막는 방법
  - 1. 이벤트 리스너에서 false 리턴

```
<a href="http://www.naver.com" onclick="return false">
        이동 안되는 링크
</a>
```

  - 2. 이벤트 객체의 preventDefault() 메소드 호출

```
<a href="http://www.naver.com" onclick="event.preventDefault();">
        이동 안되는 링크
</a>
```

- 이벤트 객체의 cancelable 프로퍼티가 true인 경우만 취소 가능

예제 9-7 이벤트의 디폴트 행동 취소


```
<!DOCTYPE html>
<html><head><meta charset="utf-8">
<title>이벤트의 디폴트 행동 취소</title>
<script>
function query() {
   let ret = confirm("네이버로 이동하시겠습니까?");
   return ret; // confirm()의 리턴 값은 true 또는 false
}

function noAction(e) {
   e.preventDefault(); // 이벤트의 디폴트 행동 강제취소
}
</script>
</head>
<body>
<h3>이벤트의 디폴트 행동 취소</h3>
<hr>
<a href="http://www.naver.com"
   onclick="return query()">
   네이버로 이동할 지 물어보는 링크</a>
<hr>
<form>
   <input type="checkbox">빵(체크 됨)<br>
   <input type="checkbox"
      onclick="noAction(event)">술(체크 안됨)
</form>
</body>
</html>
```

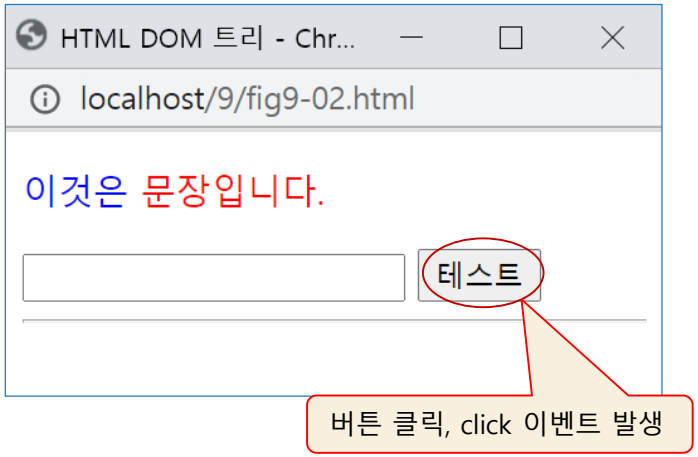
이벤트 흐름

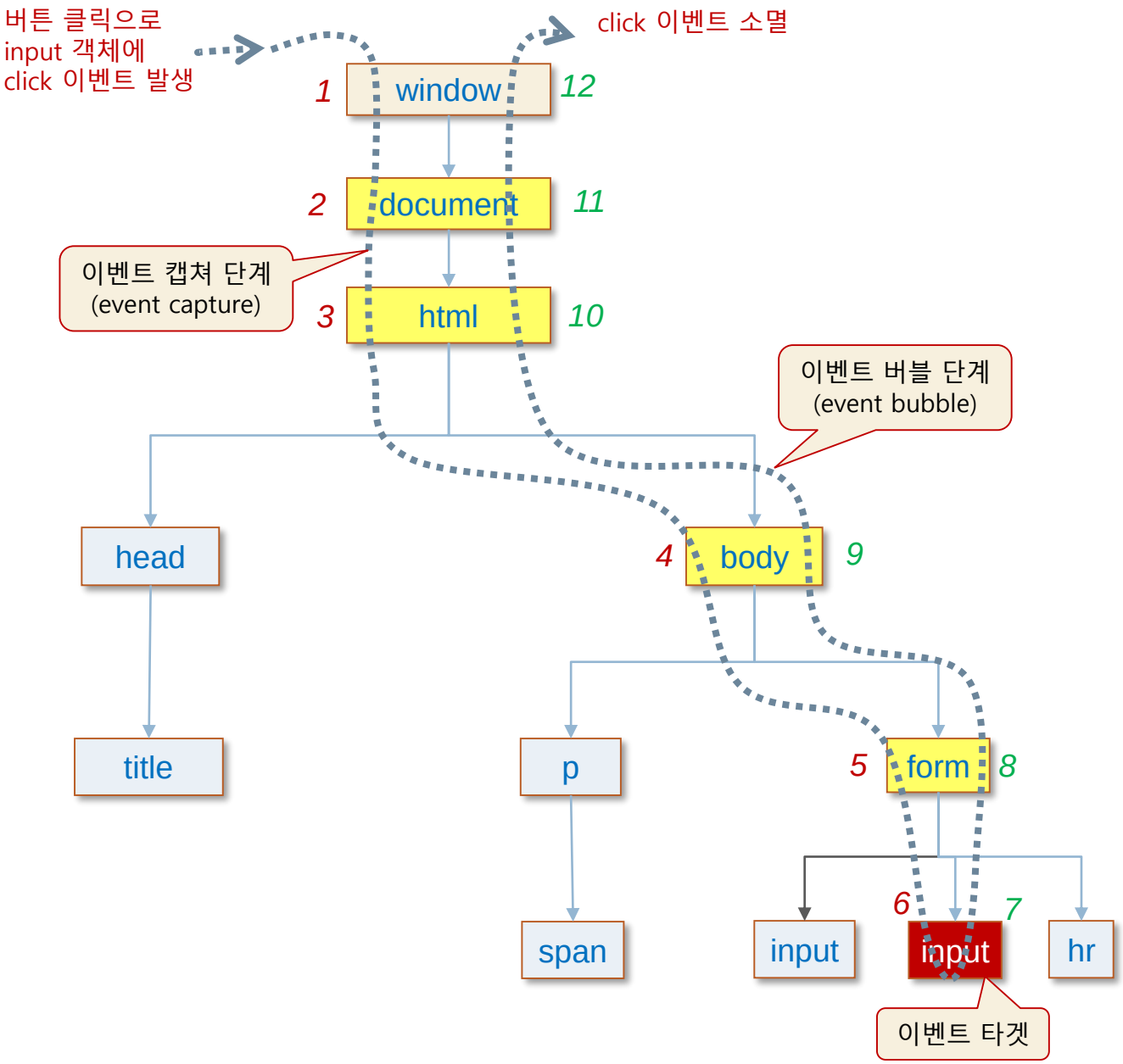
- 이벤트 흐름이란?
  - 이벤트가 발생하면 window 객체에 먼저 도달하고, DOM 트리를 따라 이벤트 타겟에 도착하고, 다시 반대 방향으로 흘러 window 객체에 도달한 다음 사라지는 과정
- 이벤트가 흘러가는 과정
  - 캡쳐 단계(capturing phase)
    - 이벤트가 window 객체에서 중간의 모든 DOM 객체를 거쳐 타겟 객체에 전달되는 과정
    - 이벤트가 거쳐가는 모든 DOM 객체(window포함)의 이벤트 리스너 실행
  - 버블 단계(bubbling phase)
    - 이벤트가 타겟에서 중간의 모든 DOM 객체를 거쳐 window 객체에 전달되는 과정
    - 이벤트가 거쳐가는 모든 DOM 객체(window포함)의 이벤트 리스너 실행
- DOM 객체에는 캡쳐 리스너와 버블 리스너 두 개 모두 작성할 수 있음

이벤트 흐름 사례

- 샘플 웹 페이지

```
<!DOCTYPE html>
<html><head><title>HTML DOM 트리</title></head>
<body>
<p style="color:blue" >이것은
    <span style="color:red">문장입니다.</span>
</p>
<form>
    <input type="text">
    <input type="button" value="테스트" id="button">
    <hr>
</form>
</body></html>
```




버튼 클릭으로
input 객체에
click 이벤트 발생
click 이벤트 소멸
1
window
12
2
document
11
이벤트 캡쳐 단계
(event capture)
3
html
10
이벤트 버블 단계
(event bubble)
head
4
body
9
title
p
5
form
8
6
7
span
input
input
hr
이벤트 타겟

캡쳐 리스너와 버블 리스너

- DOM 객체의 이벤트 리스너
  - 캡쳐 리스너와 버블 리스너를 모두 소유 가능
    - 이벤트 리스너 등록 시, 캡쳐 리스너인지 버블 리스너인지 구분
- 캡쳐 리스너와 버블 리스너 등록
  - addEventListener()의 3 번째 매개 변수 이용
    - true이면 캡쳐 리스너, false이면 버블 리스너
    - 생략되면 false로 처리

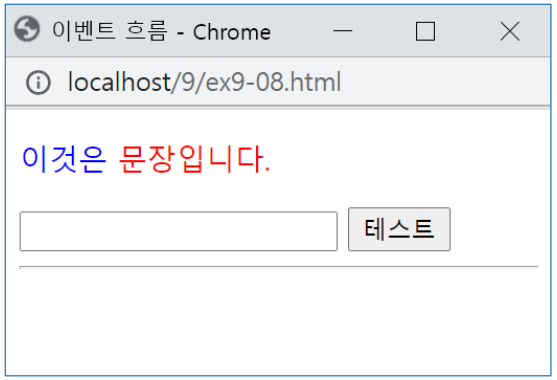
```
let b = document.getElementById("button");
b.addEventListener("click", capFunc, true); // 캡쳐 단계에서 capFunc() 실행
b.addEventListener("click", bubbleFunc, false); // 버블 단계에서 bubbleFunc() 실행
```

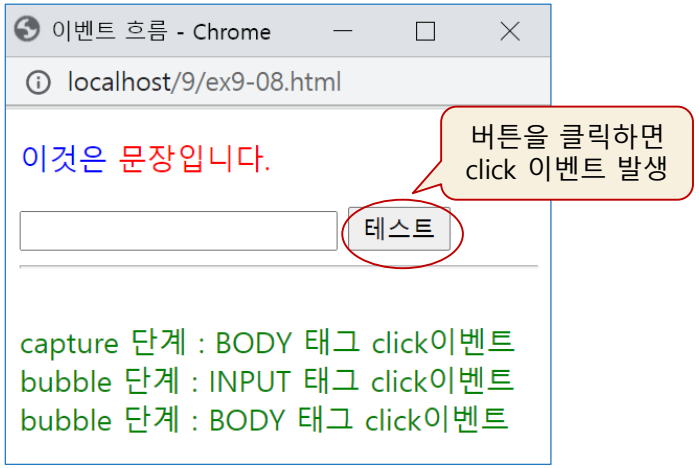
  - 다른 방법의 이벤트 리스너 등록의 경우
    - 버블 리스너로 자동 등록
    - 예)

```
obj.onclick = function(e) { // 버블 리스너도 작동
        ...
}
```

예제 9-8 이벤트 흐름

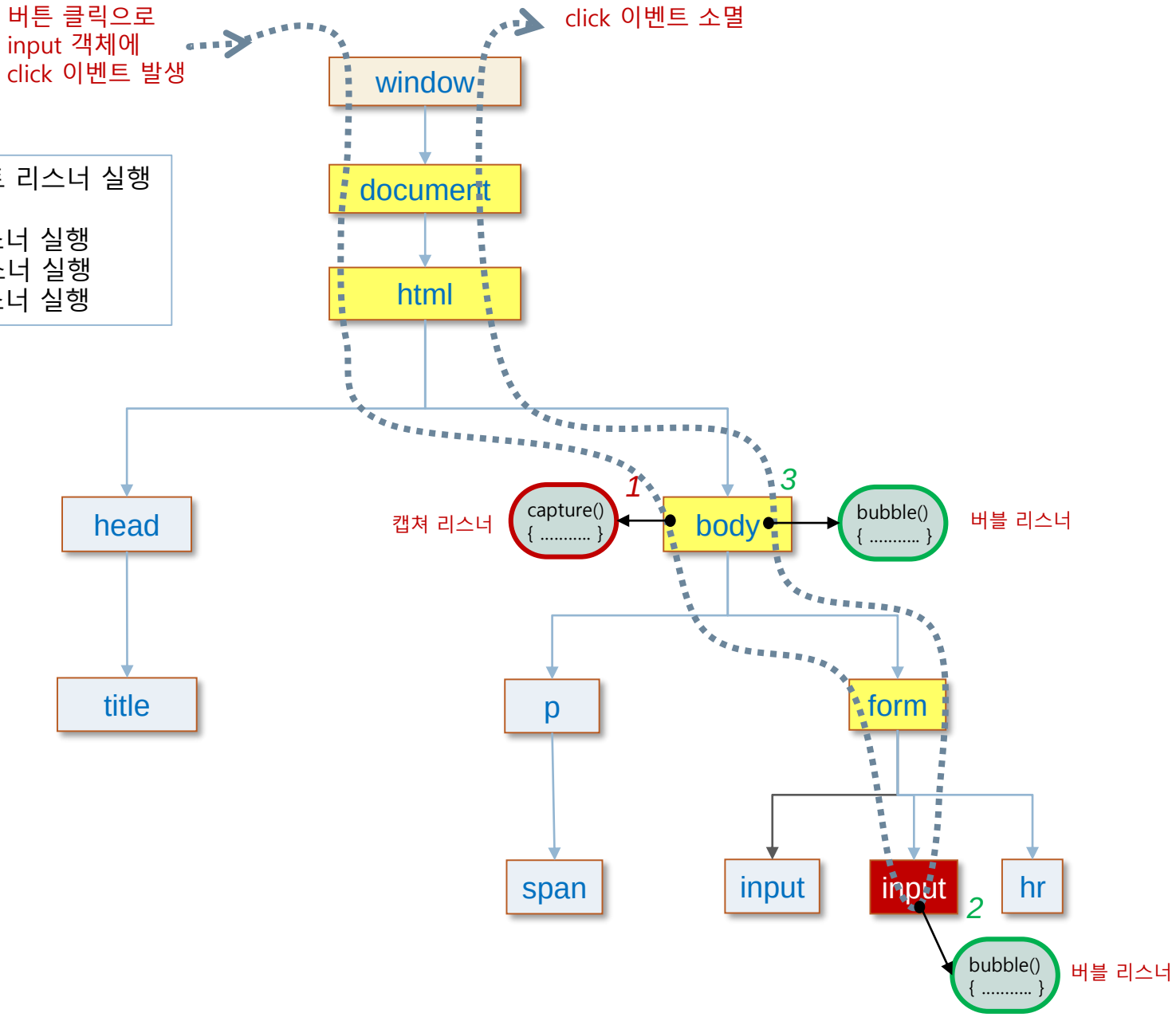

```
<!DOCTYPE html>
<html><head><meta charset="utf-8"><title>이벤트 흐름</title></head>
<body>
<p style="color:blue">이것은
    <span style="color:red" id="span">문장입니다.
    </span>
</p>
<form>
    <input type="text" name="s">
    <input type="button" value="테스트" id="button">
    <hr>
</form>
<div id="div" style="color:green"></div>
<script>
let div = document.getElementById("div"); // 이벤트 메시지 출력 공간
let button = document.getElementById("button");

// body 객체에 캡쳐 리스너 등록
document.body.addEventListener("click", capture, true); // 캡쳐 단계(1)

// 타겟 객체에 버블 리스너 등록
button.addEventListener("click", bubble, false); // 버블 단계(2)

// body 객체에 버블 리스너 등록
document.body.addEventListener("click", bubble, false); // 버블 단계(3)

function capture(e) { // e는 이벤트 객체
    let obj = e.currentTarget; // 현재 이벤트를 받은  DOM 객체
    let tagName = obj.tagName; // 태그 이름
    div.innerHTML += "<br>capture 단계 : " + tagName + " 태그 " + e.type + "이벤트";
}

function bubble(e) { // e는 이벤트 객체
    let obj = e.currentTarget; // 현재 이벤트를 받은  DOM 객체
    let tagName = obj.tagName; // 태그 이름
    div.innerHTML += "<br>bubble 단계 : " + tagName + " 태그 " + e.type + "이벤트";
}
</script>
</body></html>
```





예제 9-8 웹 페이지의 이벤트 리스너 실행

1. <body> 태그의 캡처 리스너 실행
2. <input> 태그의 버블 리스너 실행
3. <body> 태그의 버블 리스너 실행



이벤트 흐름을 중단시킬 수 있는가? YES

- 이벤트 객체의 stopPropagation() 호출
  - event.stopPropagation(); // event가 이벤트 객체일 때

마우스 핸들링

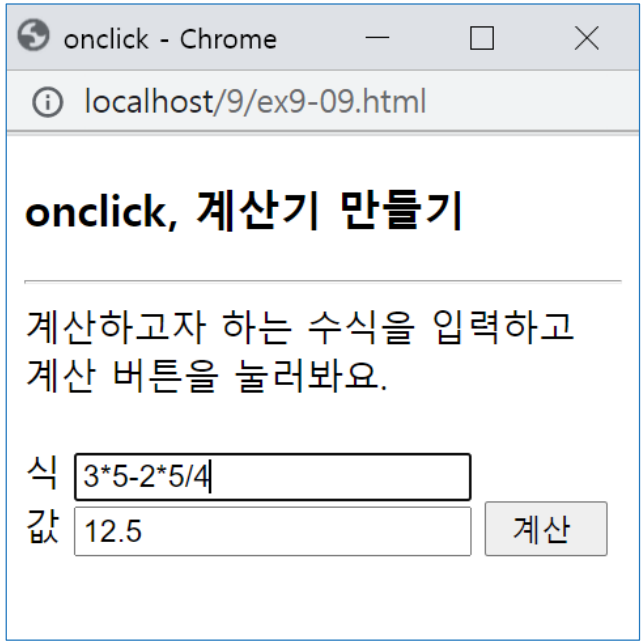
- 마우스 이벤트 객체의 프로퍼티

프로퍼티	
x, y	(x, y)는 타겟 객체의 부모 객체 내에서의 마우스 좌표
clientX, clientY	(clientX, clientY)는 브라우저 윈도우의 문서출력 영역 내에서의 마우스의 좌표
screenX, screenY	(screenX, screenY)는 스크린을 기준으로 한 마우스 좌표
offsetX, offsetY	(offsetX, offsetY)는 타겟 객체 내에서의 마우스 좌표
button	눌러진 마우스 버튼 • 0 : 아무 버튼도 눌러지지 않았음 • 1 : 왼쪽 버튼이 눌러졌음 • 2 : 오른쪽 버튼이 눌러졌음 • 3 : 왼쪽, 오른쪽 버튼이 모두 눌러졌음 • 4 : 중간 버튼이 눌러졌음
wheelDelta	마우스 휠이 구른 방향 • 양수 : 위쪽으로 굴린 경우(실제 wheelDelta 값은 120) • 음수 : 아래쪽으로 굴린 경우(실제 wheelDelta 값은 -120)

- onclick
  - HTML 태그가 클릭될 때
- ondblclick
  - HTML 태그가 더블클릭될 때

예제 9-9 onclick 리스너로 계산기 만들기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>onclick</title>
<script>
function calculate() {
    let exp = document.getElementById("exp");
    let result = document.getElementById("result");
    result.value = eval(exp.value);
}
</script>
</head>
<body >
<h3> onclick, 계산기 만들기</h3>
<hr>
계산하고자 하는 수식을
입력하고 계산 버튼을 눌러봐요.'
<br><br>
<form>
식 <input type="text" id="exp" value=""><br>
값 <input type="text" id ="result">
<input type="button" value=" 계산  "
       onclick="calculate()">
</form>
</body>
</html>
```



여러 마우스 관련 이벤트 리스너

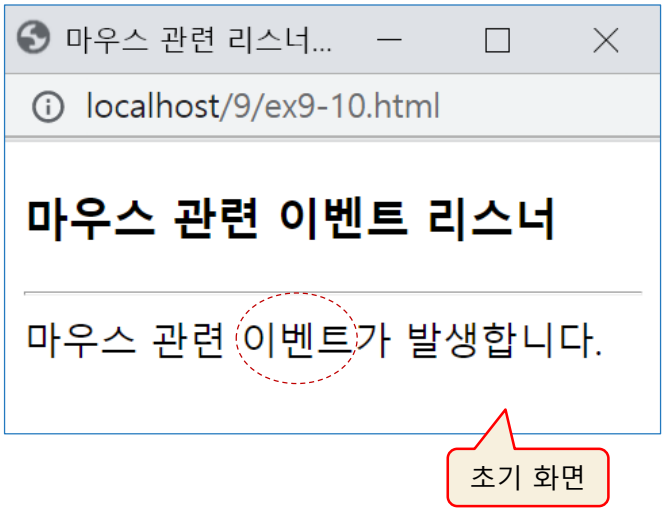
- 마우스 관련 이벤트 리스너 호출 경우
  - onmousedown : 마우스 버튼을 누르는 순간
  - onmouseup : 눌러진 버튼이 놓여지는 순간
  - onmouseover : 마우스가 태그 위로 올라오는 순간. 자식 영역 포함
  - onmouseout : 마우스가 태그 위로 올라오는 순간. 자식 영역 포함
  - onmouseenter : 마우스가 태그 위로 올라오는 순간. 버블 단계 없음
  - onmouseleave : 마우스가 태그 위로 올라오는 순간. 버블 단계 없음
  - onwheel : HTML 태그에 마우스 휠이 구르는 동안 계속 호출
    - 위쪽으로 굴린 경우 : *wheelDelta* 프로퍼티 값 양수(120)
    - 아래쪽으로 굴린 경우 : *wheelDelta* 프로퍼티 값 양수(-120)

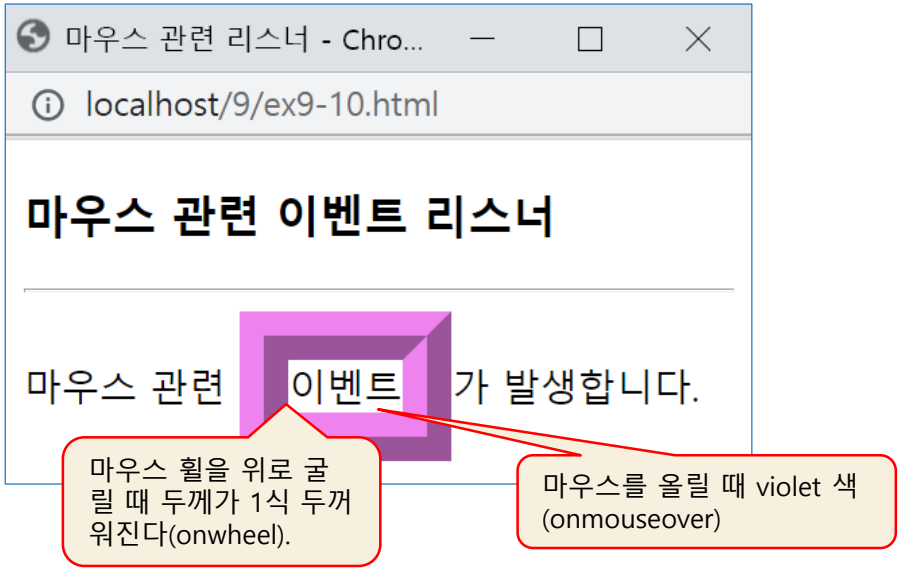
```
obj.onwheel = function (e) {
  if(e.wheelDelta < 0) { // 아래쪽으로 휠을 굴린 경우
    ...
  }
  else { // 위쪽으로 휠을 굴린 경우
    ...
  }
};
```

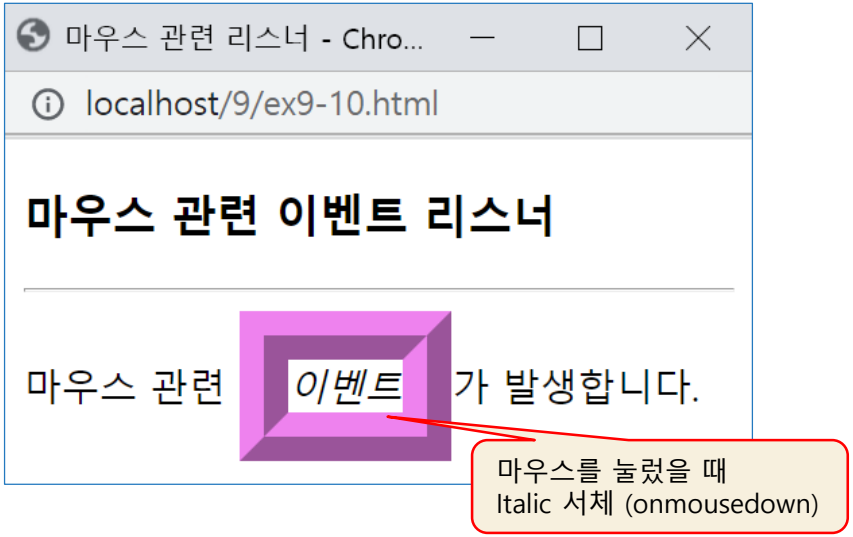
예제 9-10 마우스 관련 이벤트 리스너

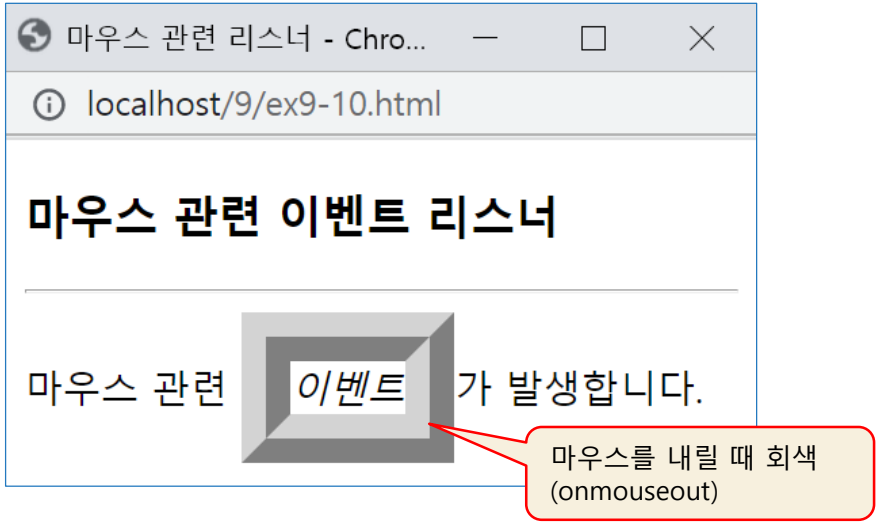

```
<!DOCTYPE html>
<html><head><meta charset="utf-8">
<title>마우스 관련 리스너</title>
<script>
let width=1; // 테두리 두께
function down(obj) {
    obj.style.fontStyle = "italic";
}
function up(obj) {
    obj.style.fontStyle = "normal";
}
function over(obj) {
    obj.style.borderColor = "violet";
    // 테두리 폭이 0일 때 색은 보이지 않는다.
}
function out(obj) {
    obj.style.borderColor = "lightgray";
}
function wheel(e, obj) { // e는 이벤트 객체
    if(e.wheelDelta < 0) { // 휠을 아래로 굴릴 때
        width--; // 폭 1 감소
        if(width < 0) width = 0; // 폭이 0보다 작아지지 않게
    }
    else // 휠을 위로 굴릴 때
        width++; // 폭 1 증가
    obj.style.borderStyle = "ridge";
    obj.style.borderWidth = width+"px";
}
</script></head>
```

```
<body >
<h3>마우스 관련 이벤트 리스너</h3>
<hr>
<div>마우스 관련
   <span onmousedown="down(this)"
         onmouseup="up(this)"
         onmouseover="over(this)"
         onmouseout="out(this)"
         onwheel="wheel(event, this)"
         style="display:inline-block">이벤트
    </span>가 발생합니다.
</div>
</body>
</html>
```


마우스 관련 리스너...
localhost/9/ex9-10.html
마우스 관련 이벤트 리스너
마우스 관련 이벤트가 발생합니다.
초기 화면

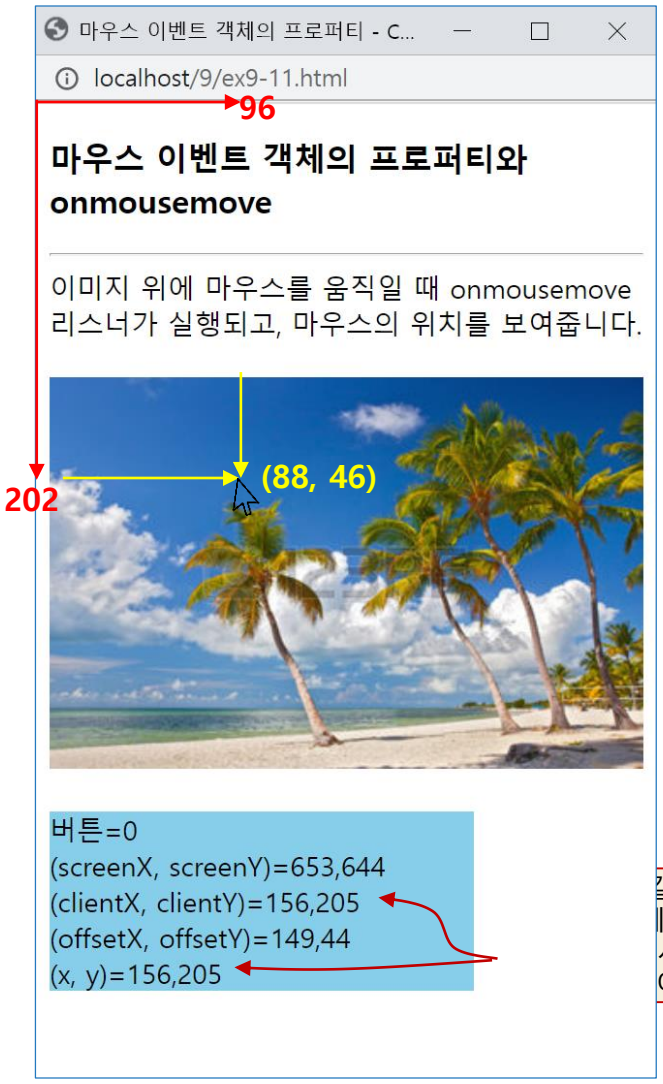

마우스 관련 리스너 - Chro...
localhost/9/ex9-10.html
마우스 관련 이벤트 리스너
마우스 관련 이벤트 가 발생합니다.
마우스 휠을 위로 굴릴 때 두께가 1식 두꺼워진다(onwheel).
마우스를 올릴 때 violet 색 (onmouseover)


마우스 관련 리스너 - Chro...
localhost/9/ex9-10.html
마우스 관련 이벤트 리스너
마우스 관련 이벤트 가 발생합니다.
마우스를 눌렀을 때 Italic 서체 (onmousedown)


마우스 관련 리스너 - Chro...
localhost/9/ex9-10.html
마우스 관련 이벤트 리스너
마우스 관련 이벤트 가 발생합니다.
마우스를 내릴 때 회색 (onmouseout)

예제 9-11 onmousemove와 마우스 위치 및 버튼


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>마우스 이벤트 객체의 프로퍼티</title>
<style>
div {
    background : skyblue;
    width : 250px;
}
</style>
</head>
<body>
<h3>마우스 이벤트 객체의 프로퍼티와 onmousemove</h3>
<hr>
이미지 위에 마우스를 움직일 때
onmousemove 리스너가 실행되고,
마우스의 위치를 보여줍니다.<br><br>
<img src="media/beach.png"
        onmousemove="where(event)"><br><br>
<div id="div"></div>
<script>
let div = document.getElementById("div");
function where(e) {
    let text = "버튼=" + e.button + "<br>";
    text += "(screenX, screenY)=" +
            e.screenX + "," + e.screenY + "<br>";
    text += "(clientX, clientY)=" +
            e.clientX + "," + e.clientY + "<br>";
    text += "(offsetX, offsetY)=" +
            e.offsetX + "," + e.offsetY + "<br>";
    text += "(x, y)=" + e.x + "," + e.y + "₩n";
    div.innerHTML = text;
}
</script>
</body></html>
```



같은 이유는
의 부모가
서, 브라우
이기 때문

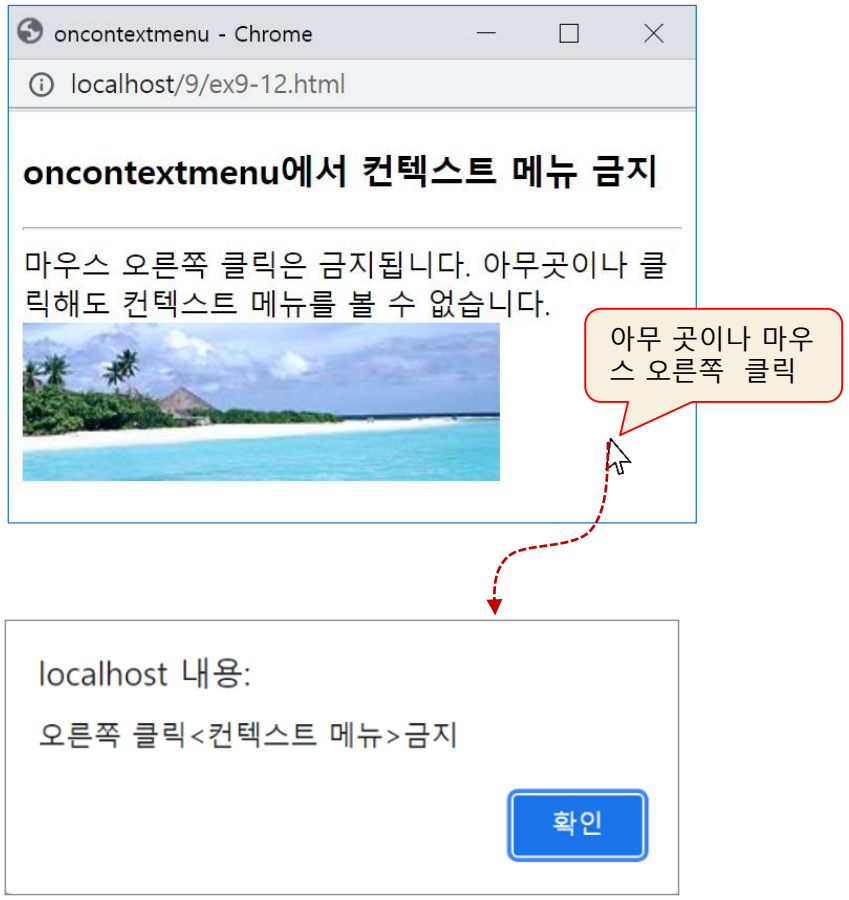
oncontextmenu

- HTML 태그 위에 마우스 오른쪽 버튼 클릭
  - 디폴트로 컨텍스트 메뉴(context menu) 출력
    - '소스 보기'나 '이미지 다운로드' 등의 메뉴
  - oncontextmenu 리스너가 먼저 호출
    - false를 리턴하면 컨텍스트 메뉴를 출력하는 디폴트 행동 취소

```
document.oncontextmenu = function () {
   ...
   return false; // 컨텍스트 메뉴 출력 금지
}
```

예제 9-12 oncontextmenu로 소스 보기나 이미지 다운로드 금지


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>oncontextmenu</title>
<script>
function hideMenu() {
    alert("오른쪽 클릭<컨텍스트 메뉴>금지");
    return false;
}
document.oncontextmenu=hideMenu;
</script>
</head>
<body>
<h3>oncontextmenu에서 컨텍스트 메뉴 금지</h3>
<hr>
마우스 오른쪽 클릭은 금지됩니다. 아무곳이나
클릭해도 컨텍스트 메뉴를 볼 수 없습니다.
<img src="media/beach2.png" alt="miami">
</body>
</html>
```



문서의 로딩 완료와 onload

- onload
  - window 객체에 발생
    - 웹 페이지의 로딩 완료시 호출되는 이벤트 리스너
  - onload 리스너 작성 방법

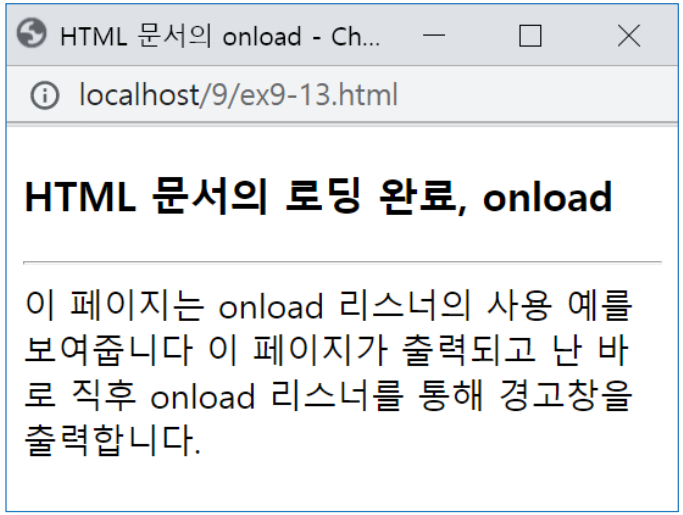
1. window.onload= function() {
   ... // 자바스크립트 코드
   };
2. <body onload="자바스크립트 코드">

예) window.onload = function() {
   alert('hello');
   }
   <body onload="alert('hello')">

예제 9-13 onload에서 사이트 이전을 알리는 공고창 출력


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>HTML 문서의 onload</title>
</head>
<body onload="alert('이 사이트는 2022년 9월1일부터 ₩
www.js.co.kr로 옮겨지게 됩니다.')">
<h3>HTML 문서의 로딩 완료, onload</h3>
<hr>
이 페이지는 onload 리스너의
사용 예를 보여줍니다
이 페이지가 출력되고 난 바로 직후
onload 리스너를 통해
경고창을 출력합니다.
</body>
</html>
```

₩는 뒤에 <enter> 키를 무시하게 만듦

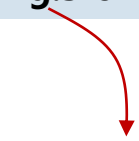


이미지 로딩 완료와 onload

- Image 객체
  - <img> 태그에 의해 생성되는 DOM 객체
  - new Image(); 자바스크립트 코드에 의해 생성되는 객체
- onload
  - 이미지의 로딩이 완료되면 Image 객체에 발생하는 이벤트
- 새로운 이미지를 로딩하는 방법

```
<img id="myImg" src="apple.png" width="..." height="...">
```

```
let myImg = document.getElementById("myImg");
myImg.src = "banana.png";
```



banana.png 이미지의 로딩이 완료된 myImg의 onload 리스너 실행

이미지 로딩시 주의할 점

- 잘못된 이미지 로딩 코드
  - 이미지를 로딩하여 이미지 폭을 알아내는 코드

```
let myImg = document.getElementById("myImg");
myImg.src = "banana.png";
let width = myImg.width;          // banana.png 이미지의 폭
```

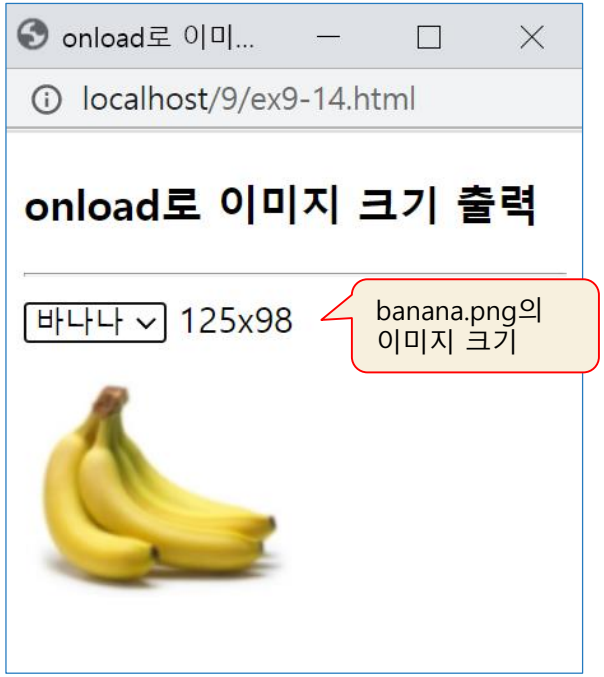
  - 문제점
    - myImg.src = "banana.png"; 실행 직후 이미지 로딩 완료되지 않음
    - let width = myImg.width; 이미지 로딩 완료 전일 때 myImg.width = 0임
- 코드 수정
  - onload 리스너에서 이미지 폭을 알아내는 코드 작성

```
let myImg = document.getElementById("myImg");
myImg.onload = function () {      // 이미지 로딩 완료 시 실행
    let width = myImg.width;      // 정확한 이미지 폭 읽기
}
myImg.src = "banana.png";         // 이미지 로딩 지시
```

width는 function() 블록 내에서만 사용할 수 있음을 주의하기 바람.

예제 9-14 onload로 이미지의 크기 알아내기


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>onload로 이미지 크기 출력</title>
<script>
function changeImage() {
    let sel = document.getElementById("sel");
    let img = document.getElementById("myImg");
    img.onload = function () {  // 이미지 크기 출력
        let mySpan = document.getElementById("mySpan");
        mySpan.innerHTML = img.width + "x" + img.height;
    }
    let index= sel.selectedIndex; // 선택된 옵션 인덱스
    img.src = sel.options[index].value; // <option>의 value 속성
}
</script>
</head>
<body onload="changeImage()">
<h3>onload로 이미지 크기 출력</h3>
<hr>
<form>
<select id="sel" onchange="changeImage()">
    <option value="images/apple.png">사과
    <option value="images/banana.png">바나나
    <option value="images/mango.png">망고
</select>
<span id="mySpan">이미지 크기</span>
</form>
<p><img id="myImg" src="media/apple.png" alt="."></p>
</body>
</html>
```



new Image()로 이미지 로딩과 출력

- 동적으로 이미지 객체 생성
  - new Image()
    - 이미지 객체가 생겼지만 화면에 출력되지 않음
- new Image()의 이미지 객체에 이미지 로딩

```
let bananaImg = new Image();        // 이미지 객체 생성
bananaImg.src = "banana.png";       // 이미지 로딩
```

- 로딩된 이미지 출력
  - <img> 태그에 할당된 브라우저 공간에 이미지 출력

```
<img id="myImg" src="apple.png" width="..." height="...">
```

```
let myImg = document.getElementById("myImg");
myImg.src = bananaImg.src;          // 이미지 출력
```

예제 9-15 new Image()로 이미지 로딩

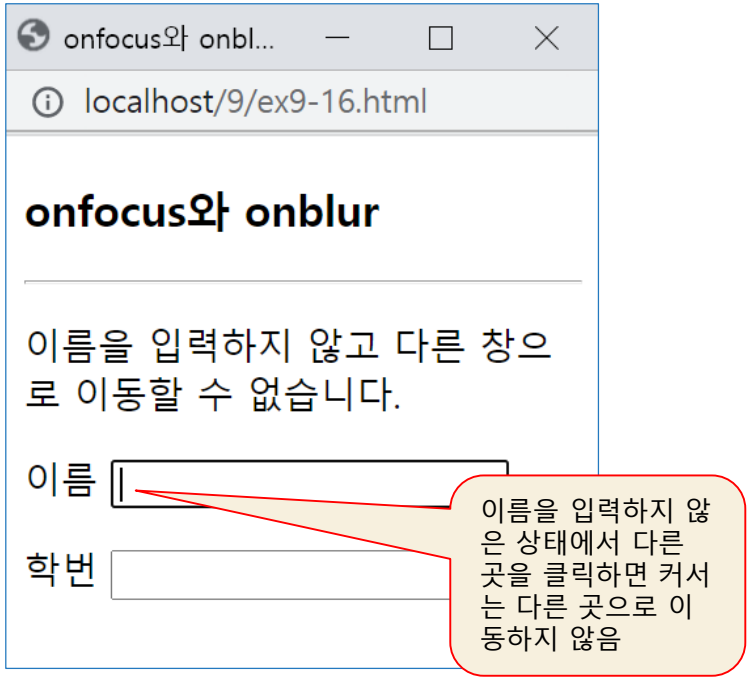

```
<!DOCTYPE html>
<html><head><meta charset="utf-8">
<title>new Image()로 이미지 로딩</title>
<script>
// 미리 로딩해둘 이미지 이름 배열
let files = ["media/Penguins.jpg",
             "media/Lighthouse.jpg",
             "media/Chrysanthemum.jpg",
             "media/Desert.jpg",
             "media/Hydrangeas.jpg",
             "media/Jellyfish.jpg",
             "media/Koala.jpg",
             "media/Tulips.jpg"];
let imgs = new Array();
for(let i=0; i<files.length; i++) {
    imgs[i] = new Image(); // 이미지 객체 생성
    imgs[i].src = files[i]; // 이미지 로딩
}

// 다음 이미지 출력
let next = 1;
function change(img) {
    img.src = imgs[next].src; // 이미지 변경
    next++; // 다음 이미지에 대한 인덱스
    next %= imgs.length; // 개수를 넘으면 처음으로
}
</script></head>
<body>
<h3>new Image()로 이미지 로딩</h3>
<hr>
이미지를 클릭하면 다음 이미지를 보여줍니다.<p>
<img style="border:20px ridge wheat"
        src="media/Penguins.jpg" alt="."  width="200" height="200"
        onclick="change(this)">
</body></html>
```

onblur와 onfocus

- 포커스
  - 포커스는 현재 키 입력에 대한 독점권
  - 브라우저는 포커스를 가지고 있는 HTML 태그 요소에 키 공급
- onblur
  - 포커스를 잃을 때 발생하는 이벤트 리스너
    - 예) 다른 HTML 요소를 클릭하면, 현재 HTML 요소는 포커스를 잃는다.
- onfocus
  - 포커스를 잃을 때 발생하는 이벤트 리스너
    - 예) 현재 HTML 요소를 클릭하면, 현재 HTML 요소가 포커스를 얻는다.

예제 9-16 onfocus와 onblur, 입력 없이 다른 창으로 갈 수 없음

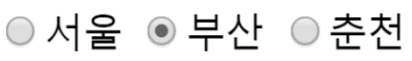

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>onfocus와 onblur</title>
<script>
function checkFilled(obj) {
    if(obj.value == "") { // obj에 입력된 것이 없다면
        obj.focus(); // obj에 다시 포커스
    }
}
</script>
</head>
<body onload="document.getElementById('name').focus();">
<h3>onfocus와 onblur</h3>
<hr>
<p>이름을 입력하지 않고 다른 창으로
이동할 수 없습니다.</p>
<form>
이름 <input type="text" id="name"
                onblur="checkFilled(this)"><p>
학번 <input type="text">
</form>
</body>
</html>
```



라디오버튼과 체크박스

- 라디오버튼 객체
  - <input type="radio">로 만들어진 라디오 버튼 DOM 객체

```
<form>
  <input type="radio" name="city" value="seoul">서울
  <input type="radio" name="city" value="busan">부산
  <input type="radio" name="city" value="chunchen">춘천
</form>
```



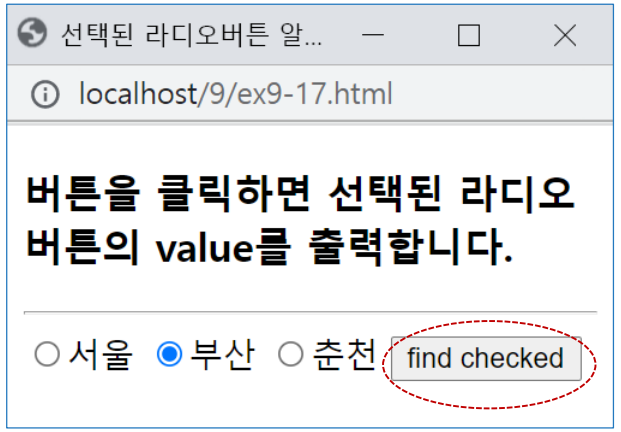
  - 라디오 버튼 객체들 알아내기

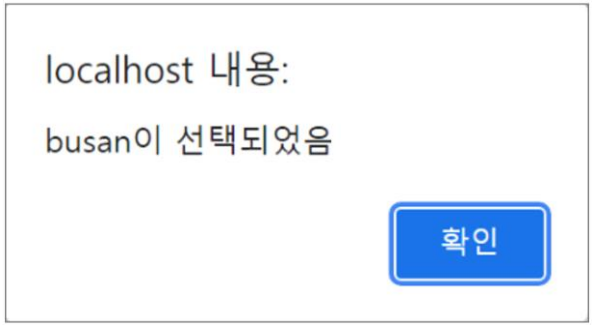
```
let kcity = document.getElementsByName("city");  // kcity[0], kcity[1], kcity[2]
```

- 체크박스 객체
  - <input type="checkbox">로 만들어진 체크박스 DOM 객체

예제 9-17 선택된 라디오버튼 알아내기

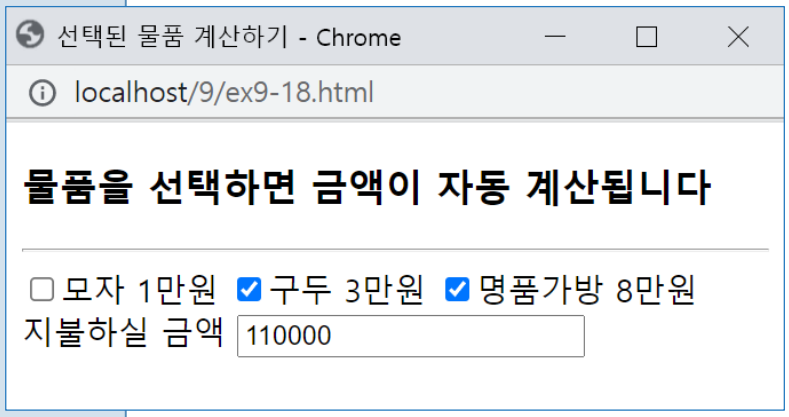

```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>선택된 라디오버튼 알아내기</title>
<script>
function findChecked() {
    let found = null;
    let kcity = document.getElementsByName("city");
    for(let i=0; i<kcity.length; i++) {
        if(kcity[i].checked == true)
            found = kcity[i];
    }
    if(found != null)
        alert(found.value + "이 선택되었음");
    else
        alert("선택된 것이 없음");
}
</script>
</head>
<body>
<h3>버튼을 클릭하면 선택된 라디오 버튼의 value를 출력합니다.</h3>
<hr>
<form>
    <input type="radio" name="city" value="seoul" checked>서울
    <input type="radio" name="city" value="busan">부산
    <input type="radio" name="city" value="chunchen">춘천
    <input type="button" value="find checked" onclick="findChecked()">
</form>
</body></html>
```





예제 9–18 체크박스로 선택한 물품 계산


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>선택된 물품 계산하기</title>
<script>
let sum=0;
function calc(cBox) {
    if(cBox.checked)
        sum += parseInt(cBox.value);
    else
        sum -= parseInt(cBox.value);
    document.getElementById("sumtext").value = sum;
}
</script>
</head>
<body>
<h3>물품을 선택하면 금액이 자동 계산됩니다</h3>
<hr>
<form>
<input type="checkbox" name="hap" value="10000"
          onclick="calc(this)">모자 1만원
<input type="checkbox" name="shose" value="30000"
          onclick="calc(this)">구두 3만원
<input type="checkbox" name="bag" value="80000"
          onclick="calc(this)">명품가방 8만원<br>
지불하실 금액 <input type="text" id="sumtext" value="0" >
</form>
</body>
</html>
```



select 객체와 onchange

- select 객체는 <select> 태그로 만들어진 콤보박스
  - option 객체는 <option>태그로 표현되는 옵션 아이템

```
<select id="fruits">
  <option value="1">딸기</option>
  <option value="2" selected>바나나</option>
  <option value="3">사과</option>
</select>
```

  - 선택된 옵션 알아내기

```
let sel = document.getElementById("fruits");
let index = sel.selectedIndex;      // index는 선택 상태의 옵션 인덱스
```

  - 옵션 선택

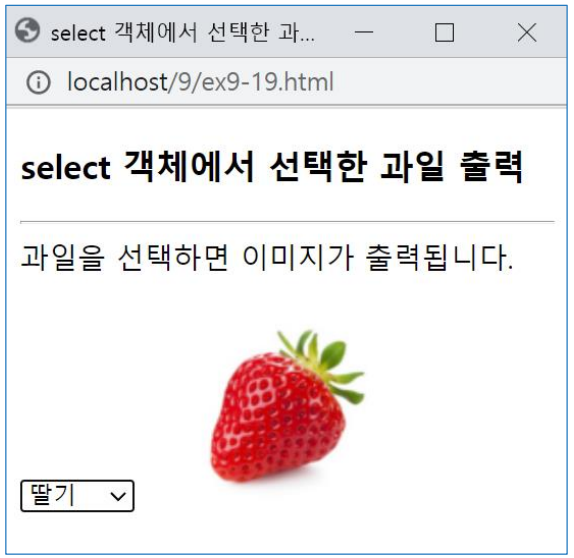
```
sel.selectedIndex = 2;                  // 3번째 옵션 "사과" 선택
sel.options[2].selected = true;         // 3번째 옵션 "사과" 선택
```

  - select와 onchange 리스너
    - 선택된 옵션이 변경되면 select 객체의 onchange 리스너 호출

```
<select id="fruits" onchange="drawImage()">...</select>
```

예제 9-19 select 객체에서 선택한 과일 출력


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>select 객체에서 선택한 과일출력</title>
<script>
function drawImage() {
    let sel = document.getElementById("fruits");
    let img = document.getElementById("fruitimage");
    img.src = sel.options[sel.selectedIndex].value;
}
</script>
</head>
<body onload="drawImage()">
<h3>select 객체에서 선택한 과일 출력</h3>
<hr>
과일을 선택하면 이미지가 출력됩니다.<p>
<form>
<select id="fruits" onchange="drawImage()">
    <option value="media/strawberry.png">딸기
    <option value="media/banana.png" selected>바나나
    <option value="media/apple.png">사과
</select>
<img id="fruitimage" src="media/banana.gif" alt="">
</form>
</body>
</html>
```

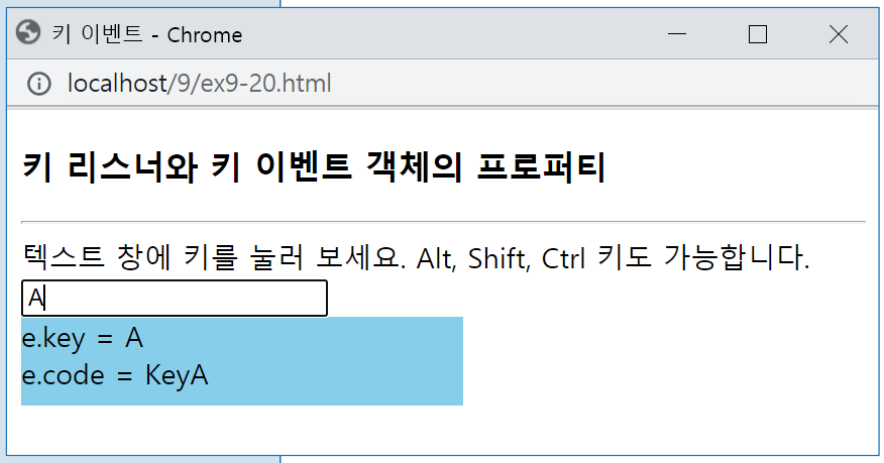


키 이벤트

- onkeydown, onkeypress, onkeyup
  - onkeydown
    - 키가 눌러지는 순간 호출. 모든 키에 대해 작동
  - onkeypress
    - 문자 키와 <Enter>, <Space>, <Esc> 키에 대해서만 눌러지는 순간에 추가 호출
      - 문자 키가 아닌 경우(<F1>, <Shift>, <PgDn>, <Del>, <Ins> 등) 호출되지 않음
  - onkeyup
    - 눌러진 키가 떼어지는 순간 호출

예제 9-20 키 이벤트 리스너와 이벤트 객체의 프로퍼티

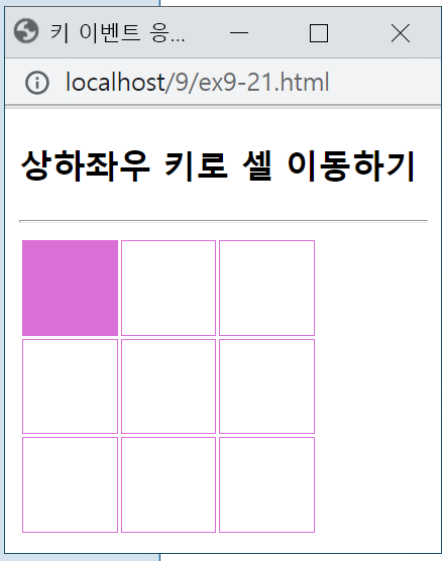

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>키 이벤트</title>
<script>
function whatKeyDown(e) {
    let str = "";
    let div = document.getElementById("div");
    div.innerHTML = ""; // div 객체 내용 초기화
    str += "e.key = " + e.key + "<br>";
    str += "e.code = " + e.code + "<br>";
    div.innerHTML = str; // div 객체에 html 문자열 출력
}
</script>
</head>
<body>
<h3>키 리스너와 키 이벤트 객체의 프로퍼티</h3>
<hr>
텍스트 창에 키를 눌러 보세요. Alt, Shift, Ctrl 키도 가능합니다.<br>
<input type="text" id="text" onkeydown="whatKeyDown(event)">
<div id="div" style="background-color:skyblue; width:250px; height:50px">
</div>
</body>
</html>
```

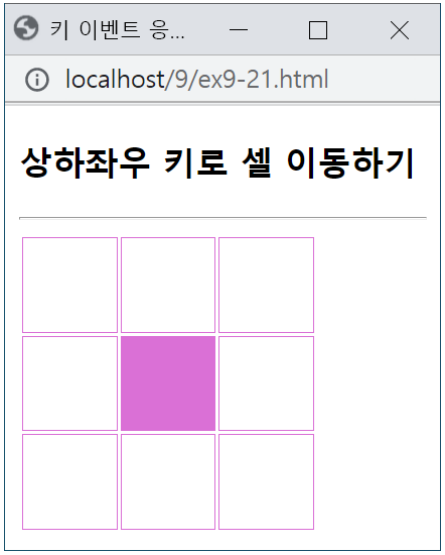


예제 9-21 키 이벤트 응용 – 상하좌우키로 셀 이동

3x3 표를 만들고 상하좌우 키를 이용하여 셀의 배경색을 바꾸면서 표의 셀 사이를 이동하는 웹 페이지를 작성하라.

```
<!DOCTYPE html>
<html><head><meta charset="utf-8"><title>키 이벤트 응용</title>
<style>
    td { width:50px; height:50px; border:1px solid orchid; }
</style>
<script>
let tds;
let prevIndex=0, index=0;
window.onload = function () { // 웹 페이지의 로딩 완료 시 실행
    tds = document.getElementsByTagName("td");
    tds[index].style.backgroundColor = "orchid";
}
window.onkeydown = function (e) {
    switch(e.key) {
        case "ArrowDown" :
            if(index/3 >= 2) return; // 맨 위 셀의 경우
            index += 3;
            break;
        case "ArrowUp" :
            if(index/3 < 1) return; // 맨 아래 셀의 경우
            index -= 3;
            break;
        case "ArrowLeft" :
            if(index%3 == 0) return; // 맨 왼쪽 셀의 경우
            index--;
            break;
        case "ArrowRight" :
            if(index%3 == 2) return; // 맨 오른쪽 셀의 경우
            index++;
            break;
    }
    tds[index].style.backgroundColor = "orchid";
    tds[prevIndex].style.backgroundColor = "white";
    prevIndex = index;
}
</script></head>
<body>
<h3>화살표 키로 셀 위로 이동하기</h3><hr>
<table>
    <tr><td></td><td></td><td></td></tr>
    <tr><td></td><td></td><td></td></tr>
    <tr><td></td><td></td><td></td></tr>
</table></body></html>
```





onreset과 onsubmit

- onreset
  - reset 버튼(<input type="reset">) 클릭 시
  - false를 리턴하면 폼이 초기화되지 않음
- onsubmit
  - submit(<input type="submit">) 버튼 클릭 시
  - false를 리턴하면 폼 전송하지 않음
- 리스너 작성
  - onreset과 onsubmit 리스너는 <form> 태그에 달아야 한다.

```
<form onreset="..." onsubmit="...">
```