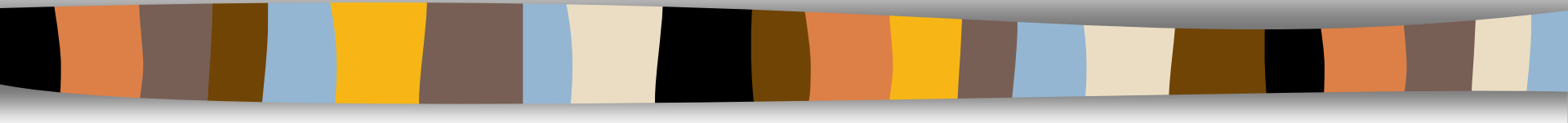


- DOM 객체의 이벤트
- BOM 객체



## - DOM 객체의 이벤트



# 이벤트 개요

3

## □ 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

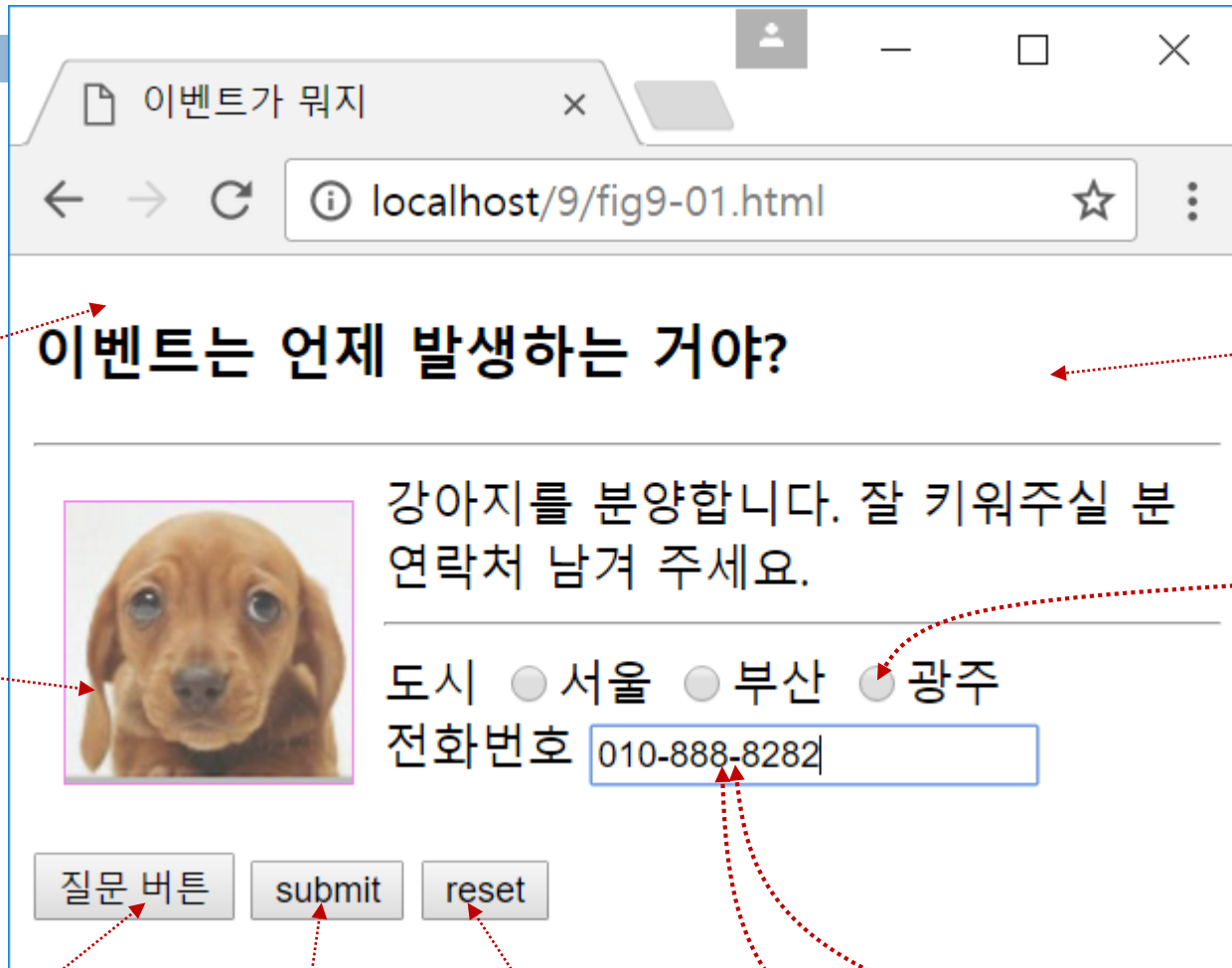
- 발생한 이벤트에 대처하기 위해 작성된 자바스크립트 코드

## □ 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

4



**load 이벤트**  
(HTML 문서 전체 로딩 완료 시)

**load 이벤트**  
(이미지의 로딩 완료 시)

**click 이벤트**  
(마우스 클릭 시)

**submit 이벤트**  
(submit 버튼 클릭 시)

**reset 이벤트**  
(reset 버튼 클릭 시)

**keypress 이벤트**  
(키를 누를 때)

**keyup 이벤트**  
(누른 키를 놓을 때)

**dblclick 이벤트**  
(마우스 더블클릭 시)

**change 이벤트**  
(라디오버튼 선택 시)

**resize 이벤트**  
(윈도우 크기 변경 시)

# 이벤트 리스너 만들기

5

## □ 3 가지 방법

- ▣ HTML 태그 내에 작성
- ▣ DOM 객체의 이벤트 리스너 프로퍼티에 작성
- ▣ DOM 객체의 `addEventListener()` 메소드 이용

## □ HTML 태그 내에 이벤트 리스너 작성

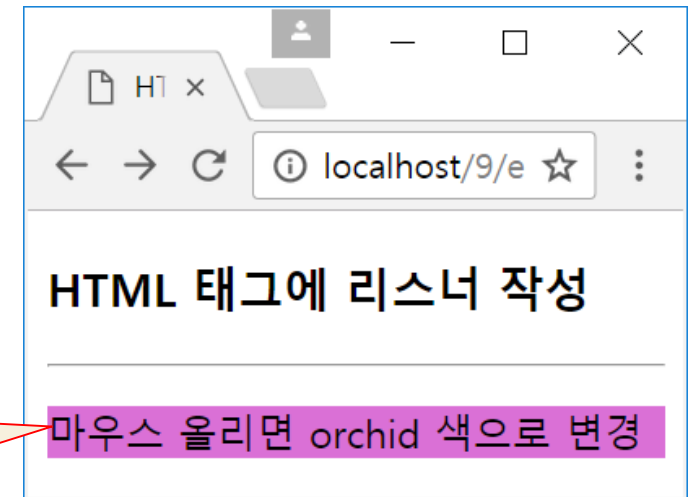
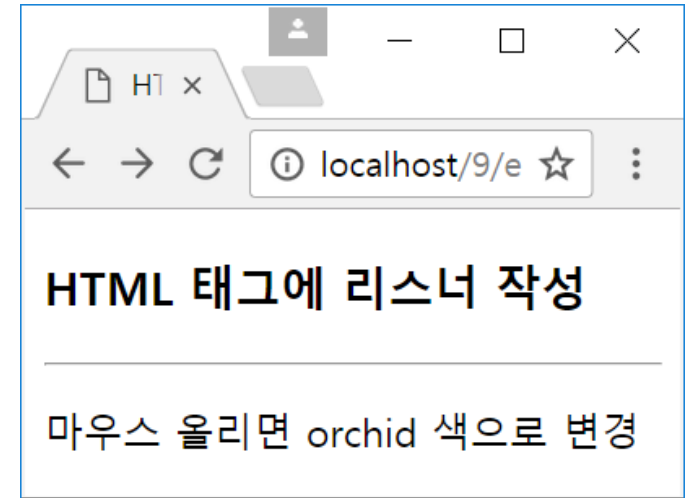
- ▣ HTML 태그의 이벤트 리스너 속성에 리스너 코드 직접 작성  
예) <p>태그에 마우스 올리면 orchid, 내리면 흰색으로 배경변경

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

# 예제 : HTML 태그 내에 이벤트 리스너 작성

6

```
<!DOCTYPE html>
<html>
<head> <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>
```



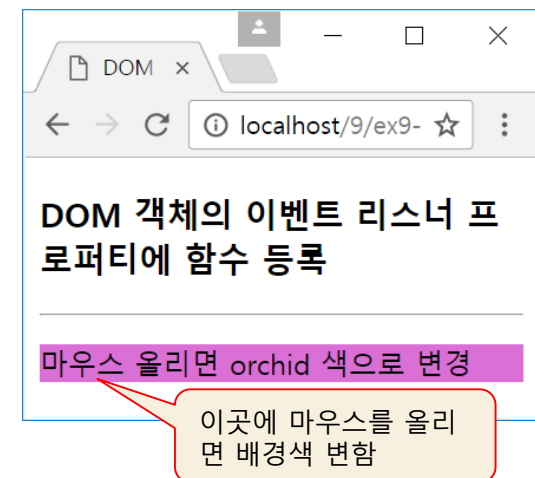
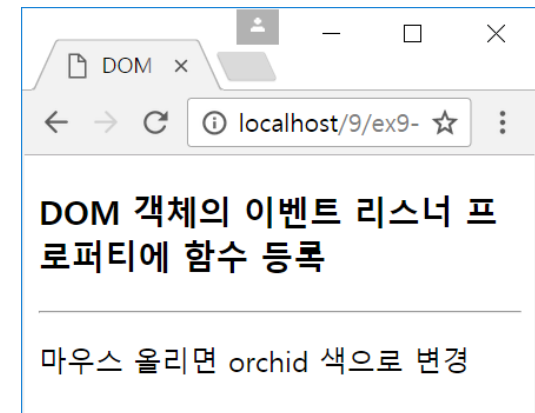
이곳에 마우스를 올리  
면 배경색 변함

마우스 올리면 orchid 색으로 변경

# 예제: DOM 객체의 이벤트 리스너 프로퍼티에 작성

7

```
<!DOCTYPE html>
<html>
<head>
<title>DOM 객체의 이벤트 리스너 프로퍼티에 함수 등록</title>
<script>
var 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

8

## □ 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 캡처 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)

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

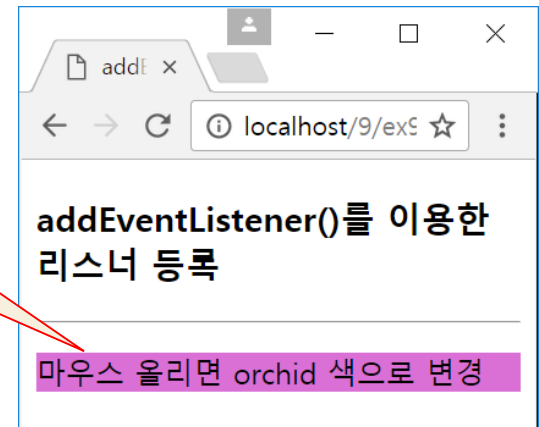
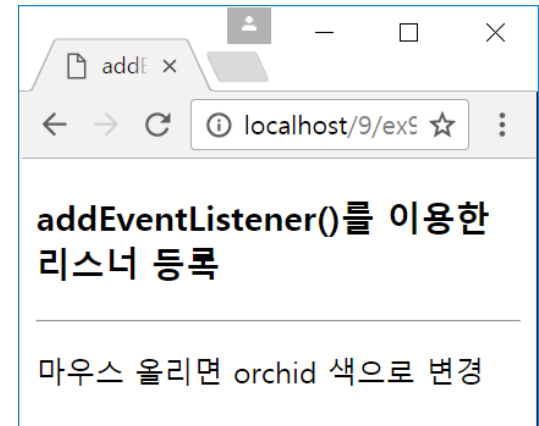


# 예제: addEventListener() 사용

9

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

이곳에 마우스를 올리면  
배경색 변함



# 익명 함수로 이벤트 리스너 작성

10

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

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

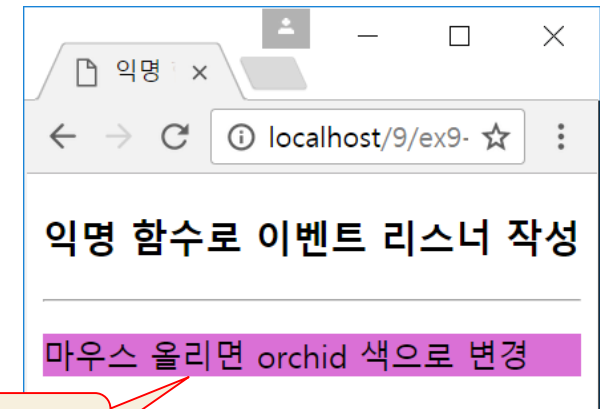
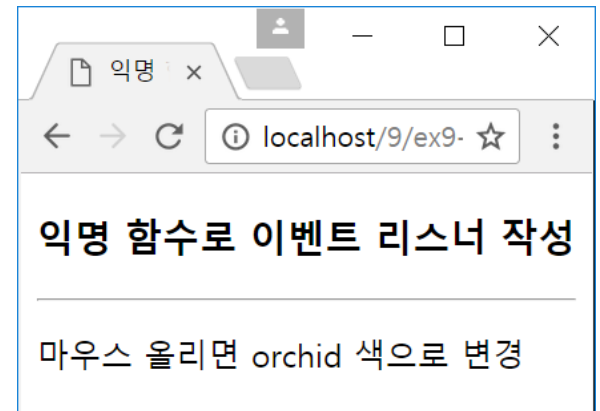
```
p.addEventListener("mouseover",  
    function () { this.style.backgroundColor = "orchid"; } // 익명 함수  
);
```

- ▣ 코드가 짧거나 한 곳에서만 사용하는 경우, 익명 함수 편리

# 예제

11

```
<!DOCTYPE html>
<html>
<head>
<title>익명 함수로 이벤트 리스너 작성</title>
<script>
var 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>
```

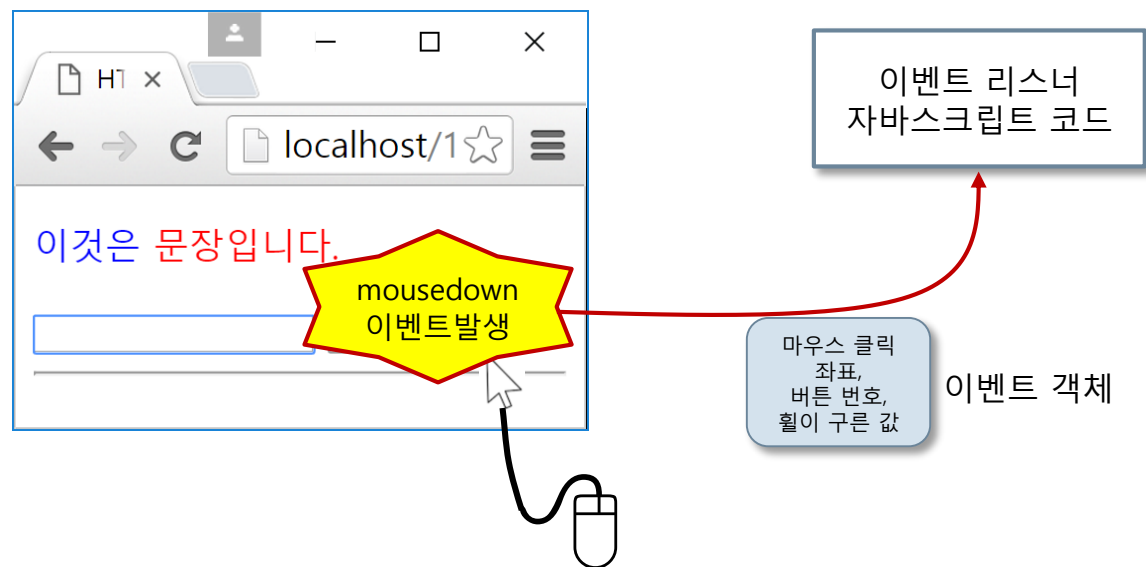


이곳에 마우스를 올리  
면 배경색 변함

# 이벤트 객체

12

- 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

13

- 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>
```

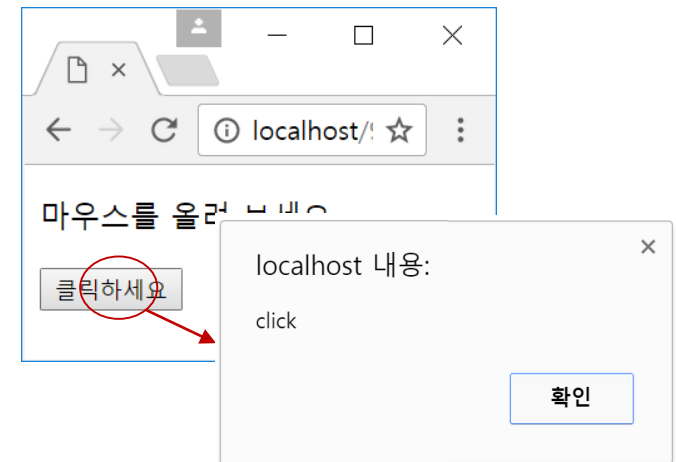
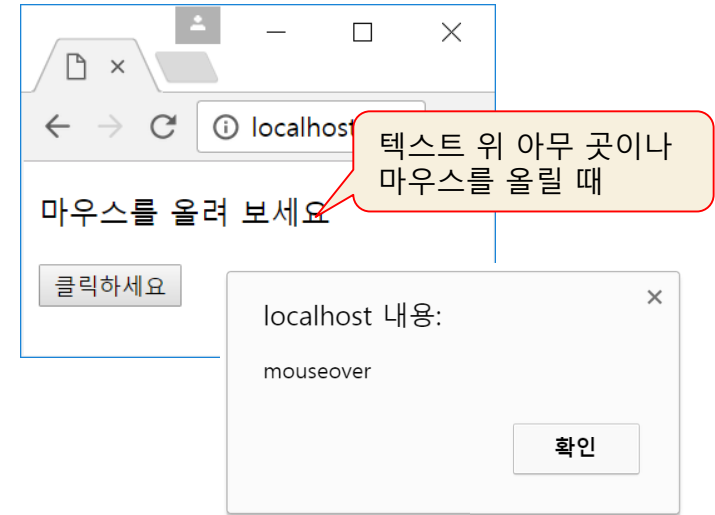
event 라는 이름으로 이벤트 객체 전달받음

# 예제

14

```
<!DOCTYPE html>
<html>
<head>
<title>이벤트 객체 전달받기</title>
</head>
<body>
<p id="p">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

15

- 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

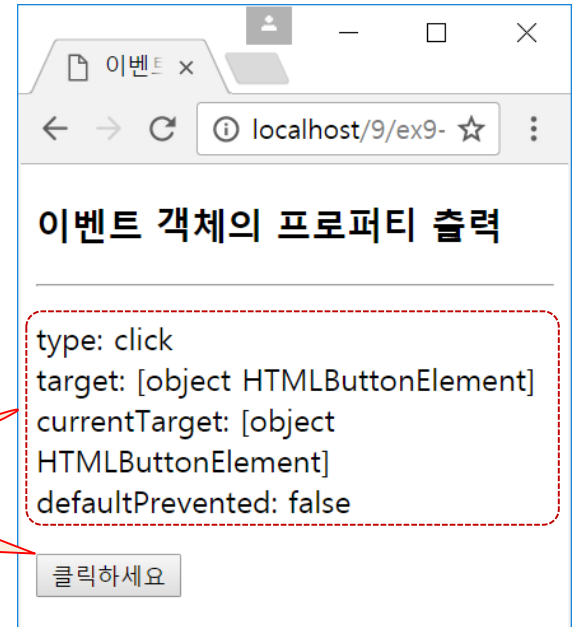
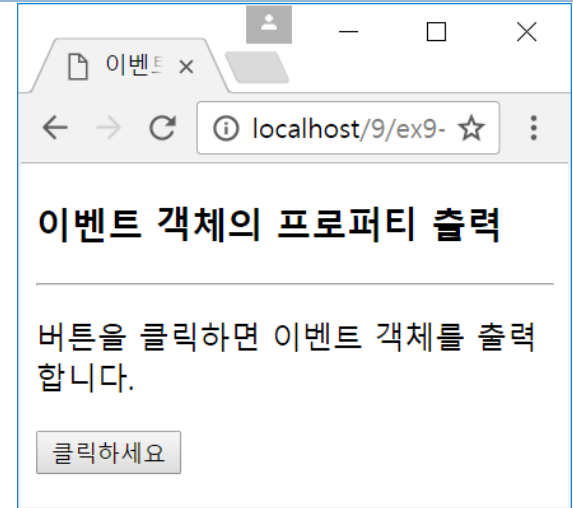
# 예제: 이벤트 객체의 프로퍼티 출력

16

```
<!DOCTYPE html>
<html>
<head><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
  var text = "type: " + e.type + "<br>"
    + "target: " + e.target + "<br>"
    + "currentTarget: " + e.currentTarget + "<br>"
    + "defaultPrevented: " + e.defaultPrevented;

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

버튼을 클릭하면 click  
이벤트 객체의 프로퍼  
티 출력





# 이벤트의 디폴트 행동 취소, preventDefault()

17

- 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

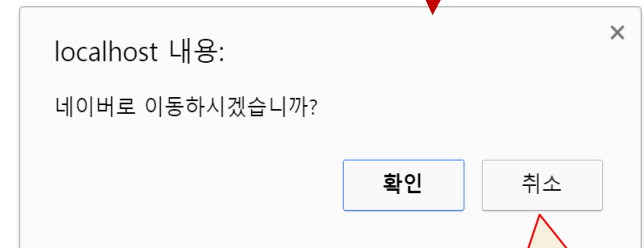
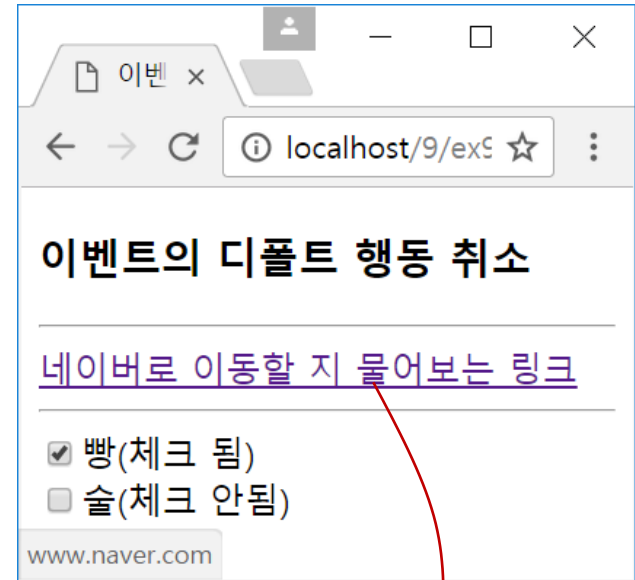
```
<a href="http://www.naver.com" onclick="event.preventDefault();">
    이동 안되는 링크
</a>
```
- 이벤트 객체의 cancelable 프로퍼티가 true인 경우만 취소 가능

# 예제: 이벤트의 디폴트 행동 취소

18

```
<!DOCTYPE html>
<html><head><title>이벤트의 디폴트 행동 취소</title>
<script>
function query() {
    var 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>
```



취소 버튼을 누르면  
네이버로 이동하지 않음

# 이벤트 흐름

19

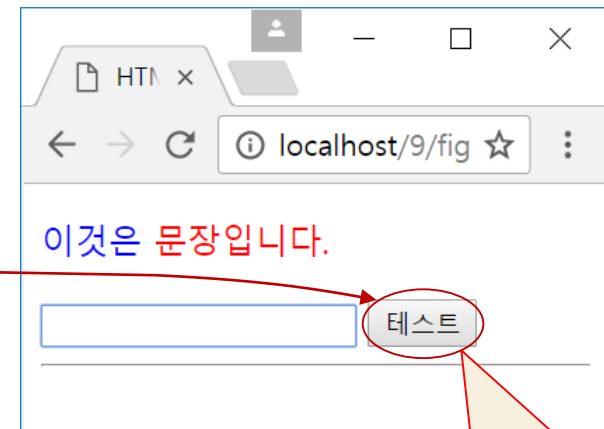
- 이벤트 흐름이란?
  - 이벤트가 발생하면 window 객체에 먼저 도달하고, DOM 트리를 따라 이벤트 타겟에 도착하고, 다시 반대 방향으로 흘러 window 객체에 도달한 다음 사라지는 과정
- 이벤트가 흘러가는 과정
  - 캡처 단계(capturing phase)
    - 이벤트가 window 객체에서 중간의 모든 DOM 객체를 거쳐 타겟 객체에 전달되는 과정
    - 이벤트가 거쳐가는 모든 DOM 객체(window포함)의 이벤트 리스너 실행
  - 버블 단계(bubbling phase)
    - 이벤트가 타겟에서 중간의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

# 이벤트 흐름 사례

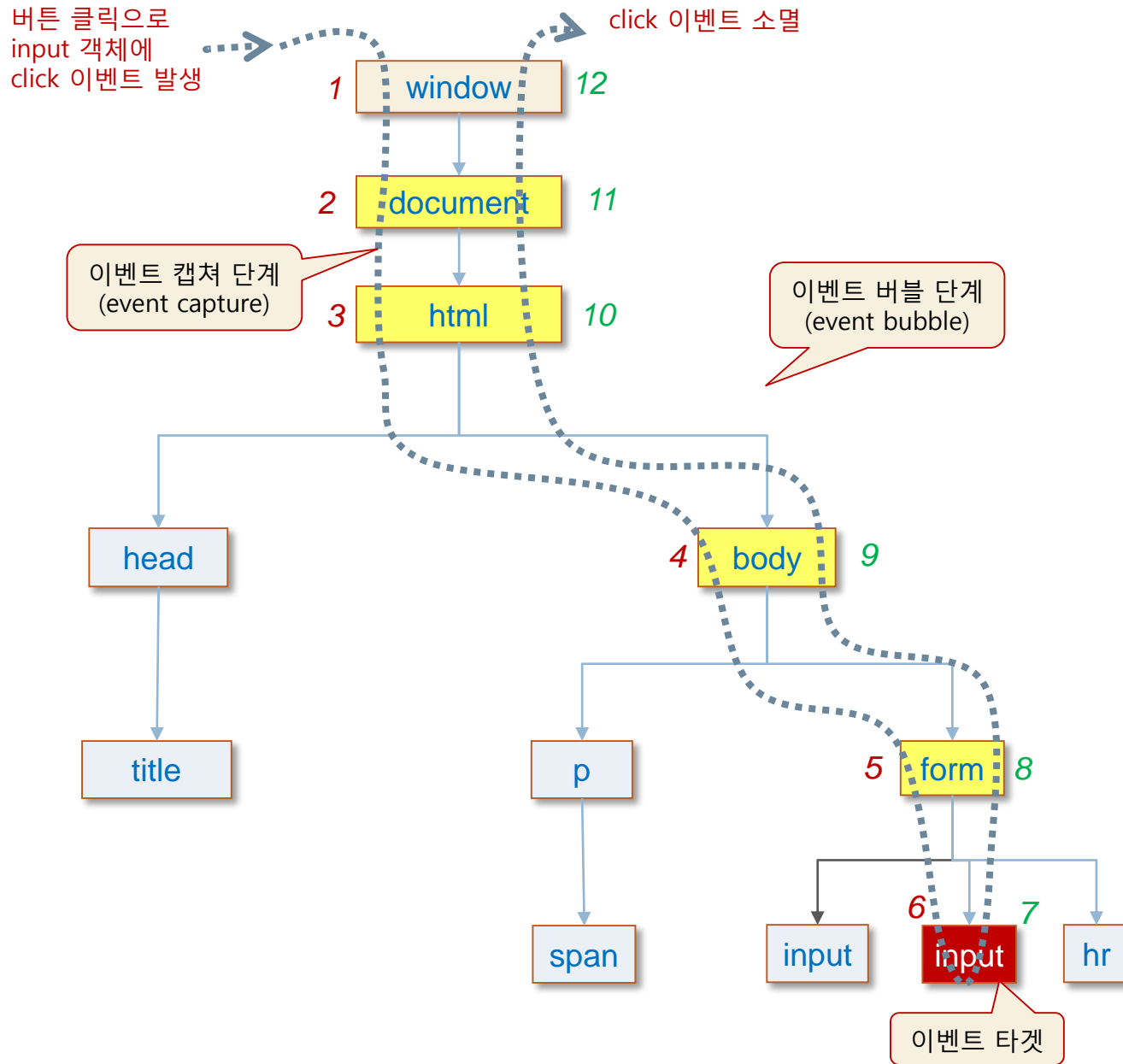
20

## □ 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 클릭, click 이벤트 발생



# 캡처 리스너와 버블 리스너

22

- DOM 객체의 이벤트 리스너
  - ▣ 캡처 리스너와 버블 리스너를 모두 소유 가능
    - 이벤트 리스너 등록 시, 캡처 리스너인지 버블 리스너인지 구분
- 캡처 리스너와 버블 리스너 등록
  - ▣ `addEventListener()`의 3 번째 매개 변수 이용
    - `true`이면 캡처 리스너, `false`이면 버블 리스너

```
var b = document.getElementById("button");  
b.addEventListener("click", capFunc, true); // 캡처 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: 이벤트 흐름

23

```
<!DOCTYPE html>
<html><head><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">
</form>
<div id="div" style="color:green"></div>
<script>
var div = document.getElementById("div"); // 이벤트 메시지 출력 공간
var button = document.getElementById("button");

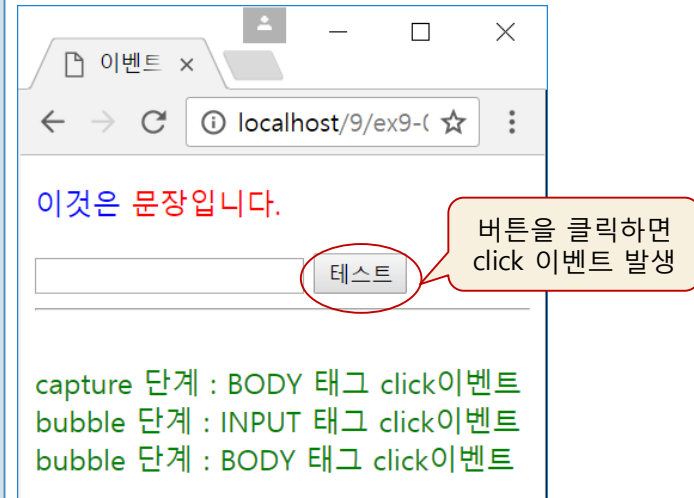
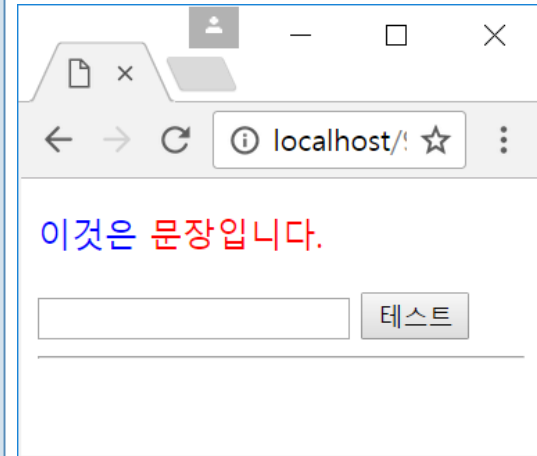
// body 객체에 캡처 리스너 등록
document.body.addEventListener("click", capture, true); // 캡처 단계(1)

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

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

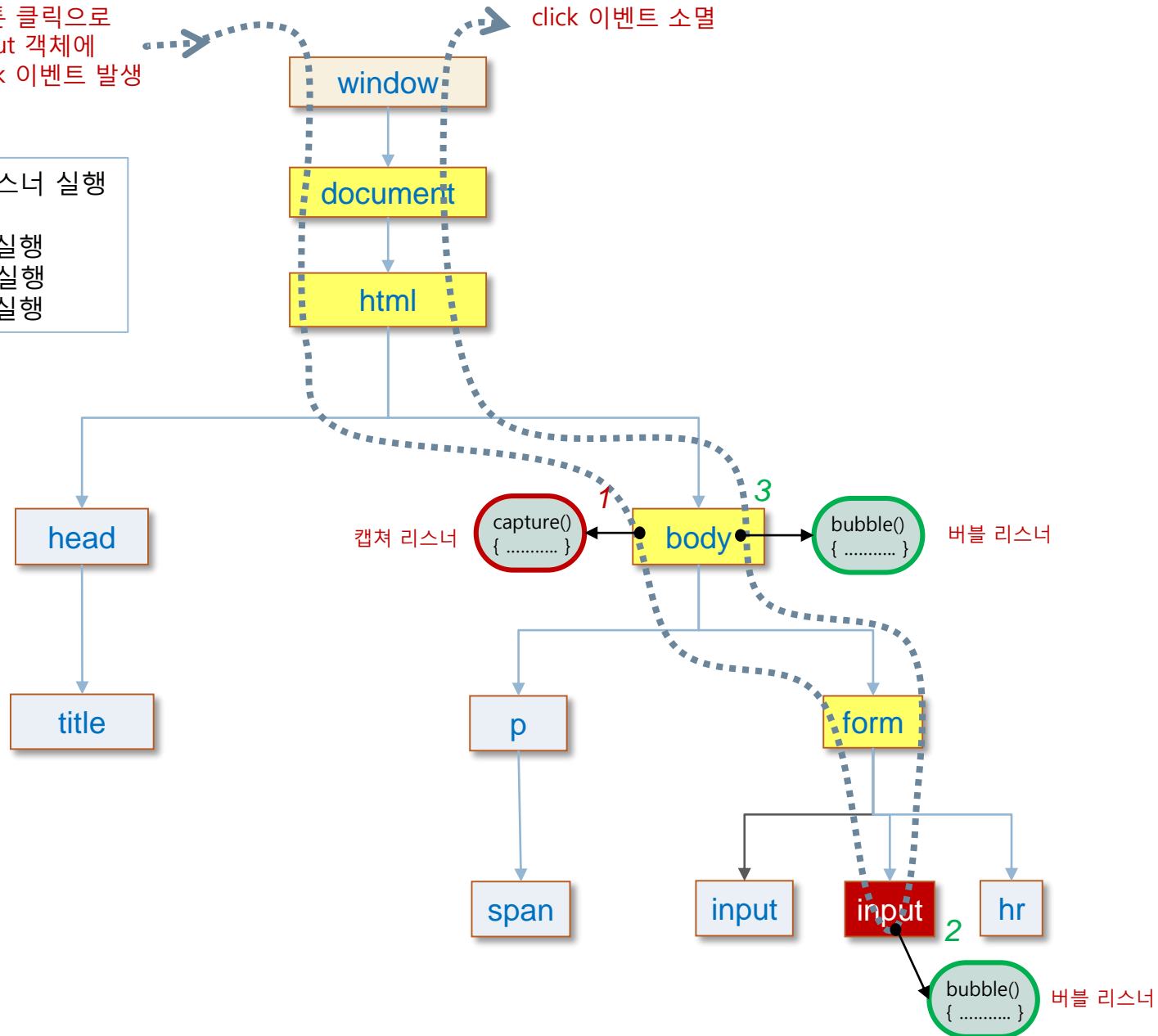


버튼 클릭으로  
input 객체에  
click 이벤트 발생

click 이벤트 소멸

### 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

25

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

# 마우스 핸들링

26

## □ 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	눌려진 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

# 여러 마우스 관련 이벤트 리스너

27

## ■ 마우스 관련 이벤트 리스너 호출 경우

- onmousedown : 마우스 버튼을 누르는 순간
- onmouseup : 눌려진 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)

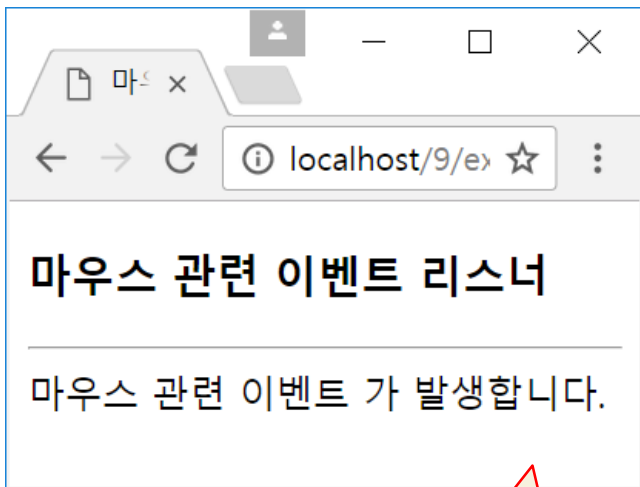
```
obj.onwheel = function (e) {  
  if(e.wheelDelta < 0) { // 아래쪽으로 휠을 굴린 경우  
    ...  
  }  
  else { // 위쪽으로 휠을 굴린 경우  
    ...  
  }  
};
```

# 예제: 마우스 관련 이벤트 리스너

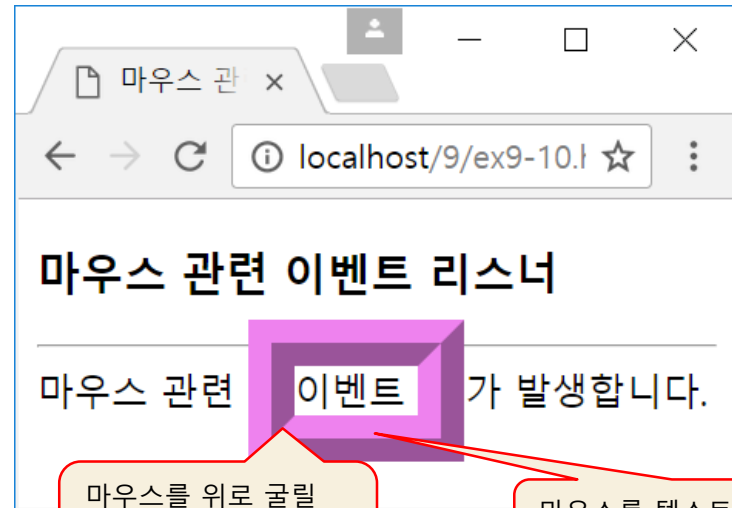
28

```
<!DOCTYPE html>
<html><head><title>마우스 관련 리스너</title>
<script>
var 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)">이벤트
    </span>가 발생합니다.
</div>
</body>
</html>
```

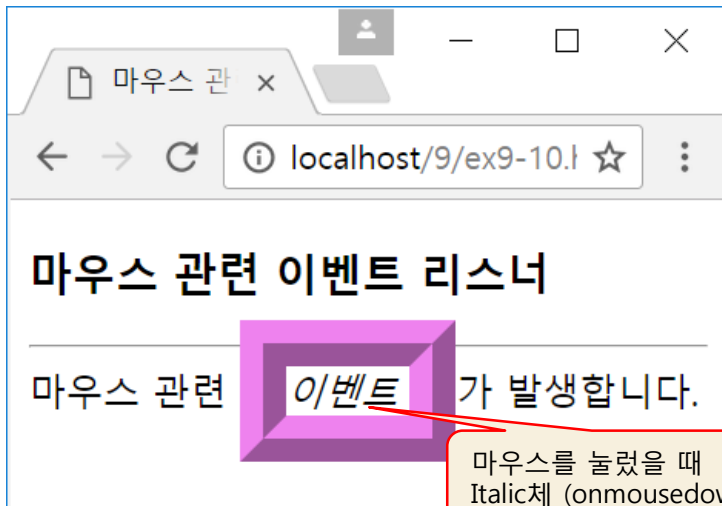


초기 화면

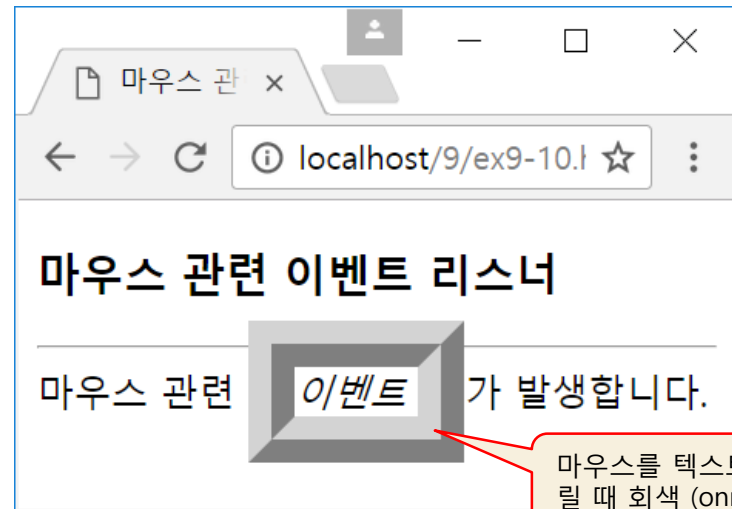


마우스를 위로 굴릴 때 두께가 1씩 두꺼워진다. (onwheel)

마우스를 텍스트위로 올릴 때 violet 색(onmouseover)



마우스를 눌렀을 때 Italic체 (onmousedown)



마우스를 텍스트에서 내릴 때 회색 (onmouseout)

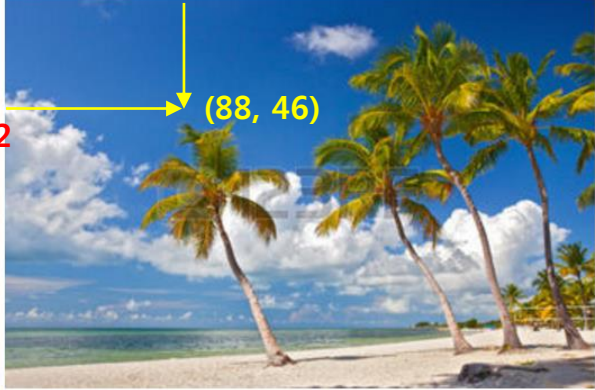
# 예제: onmousemove와 마우스 위치 및 버튼

30

```
<!DOCTYPE html>
<html><head><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.<p>
<p>
<div id="div"></div>
<script>
var div = document.getElementById("div");
function where(e) {
  var 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>
```

마우스 이벤트 객체의 프로퍼티와 onmousemove

이미지 위에 마우스를 움직일 때 onmousemove 리스너가 실행되고, 마우스의 위치를 보여줍니다.



버튼=0  
(screenX, screenY)=408,478  
(clientX, clientY)=96,202  
(offsetX, offsetY)=88,46  
(x, y)=96,202

두 좌표가 같은 이유는 <img>객체의 부모가 <body>로서, 브라우저 윈도우이기 때문

# oncontextmenu

31

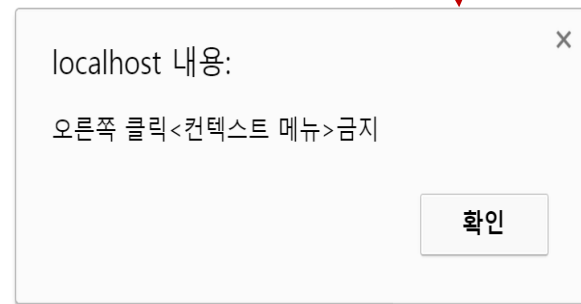
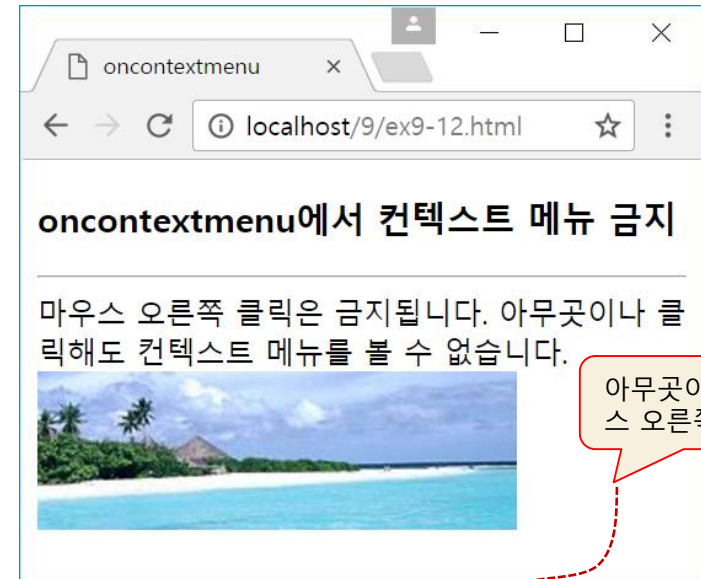
- HTML 태그 위에 마우스 오른쪽 버튼 클릭
  - ▣ 디폴트로 컨텍스트 메뉴(context menu) 출력
    - ‘소스 보기’나 ‘이미지 다운로드’ 등의 메뉴
  - ▣ oncontextmenu 리스너가 먼저 호출
    - false를 리턴하면 컨텍스트 메뉴를 출력하는 디폴트 행동 취소

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

# 예제: oncontextmenu로 소스 보기나 이미지 다운로드 금지

32

```
<!DOCTYPE html>
<html>
<head> <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.
</body>
</html>
```





# 문서의 로딩 완료와 onload

33

## □ onload

### ▣ window 객체에 발생

- 웹 페이지의 로딩 완료시 호출되는 이벤트 리스너

### ▣ onload 리스너 작성 방법

1. `window.onload="alert('onload');";`
2. `<body onload="alert('onload');">`

이 둘은 같은 표현임.

<body>에 onload를 달인 window 객체에 load 이벤트가 전달됨

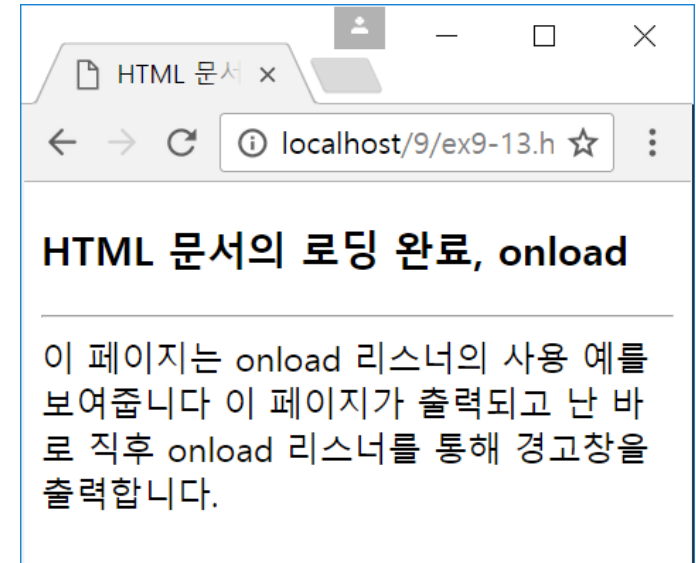
\* document.onload는 최근에 와서 많은 브라우저에서 작동하지 않음

# 예제:

34

```
<!DOCTYPE html>
<html>
<head><title>HTML 문서의 onload</title>
</head>
<body onload="alert('이 사이트는 2017년 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 만들



localhost 내용:

이 사이트는 2017년 9월1일부터 www.js.co.kr로 옮겨지게 됩니다.

확인

# 이미지 로딩 완료와 onload


35

- 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

```

```

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



banana.png 이미지의 로딩이 완료된 myImg의 onload 리스너 실행

# onblur와 onfocus

36

## □ 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.

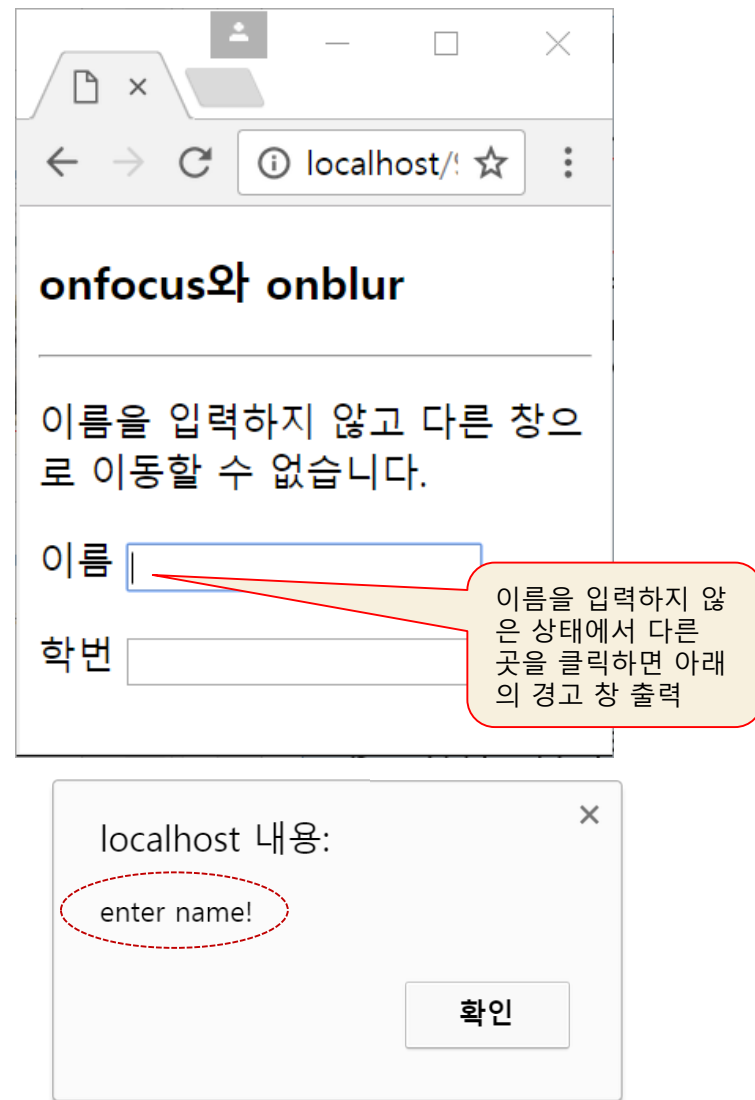
## □ onfocus

- ▣ 포커스를 얻을 때 발생하는 이벤트 리스너
  - 예) 현재 HTML 요소를 클릭하면, 현재 HTML 요소가 포커스를 얻는다.

# 예제: onfocus와 onblur, 입력 없이 다른 창으로 갈 수 없음

37

```
<!DOCTYPE html>
<html>
<head> <title>onfocus와 onblur</title>
<script>
function checkFilled(obj) {
  if(obj.value == "") {
    alert("enter name!");
    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

38

## □ 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>
```

☐ 서울 ☒ 부산 ☐ 춘천

- ▣ 라디오 버튼 객체들 알아내기

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

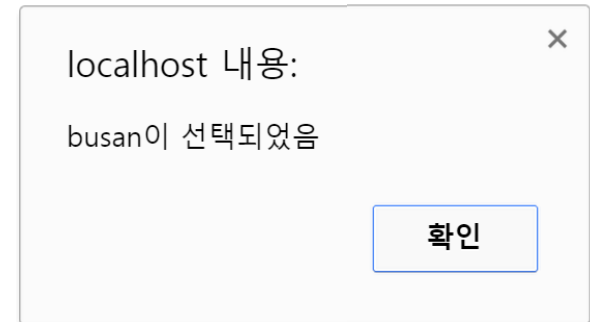
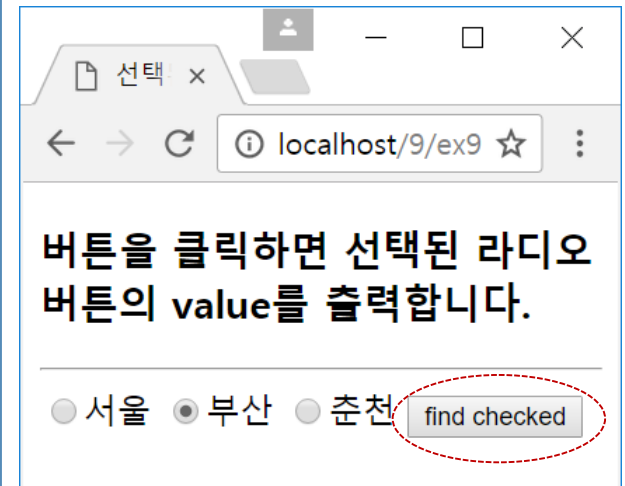
## □ 체크박스 객체

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

# 예제: 선택된 라디오버튼 알아내기

39

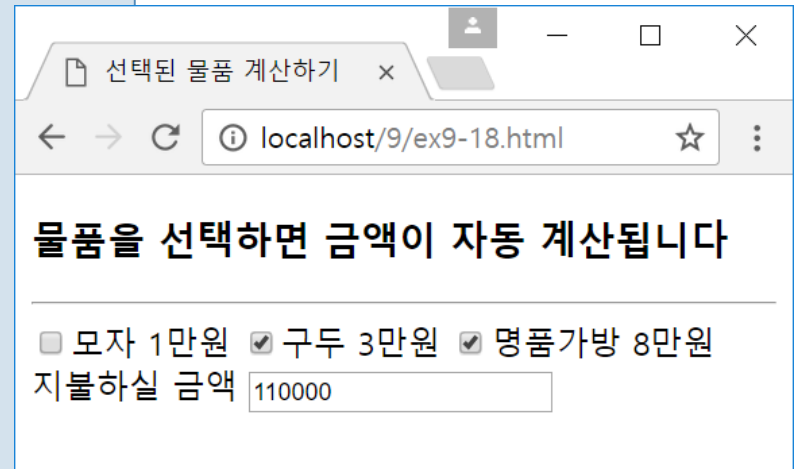
```
<!DOCTYPE html>
<html>
<head> <title>선택된 라디오버튼 알아내기</title>
<script>
function findChecked() {
    var found = null;
    var kcity = document.getElementsByName("city");
    for(var i=0; i<kcity.length; i++) {
        if(kcity[i].checked == true)
            found = kcity[i];
    }
    if(found != 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: 체크박스로 선택한 물품 계산

40

```
<!DOCTYPE html>
<html>
<head> <title>선택된 물품 계산하기</title>
<script>
var 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="cap" 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>
```



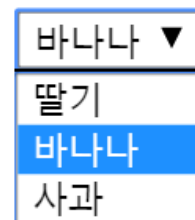


# select 객체와 onchange

41

- 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

```
var sel = document.getElementById("fruits");  
var 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

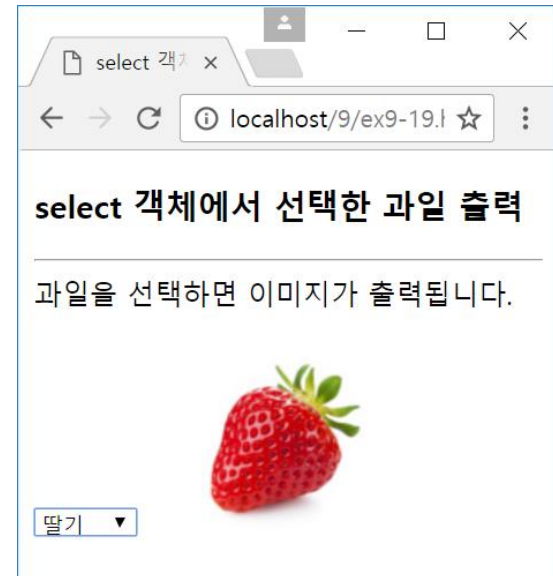
```
<select id="fruits" onchange="drawImage()">...</select>
```

# 예제: select 객체에서 선택한 과일 출력

42

```
<!DOCTYPE html>
<html>
<head>
<title>select 객체에서 선택한 과일출력</title>
<script>
function drawImage() {
    var sel = document.getElementById("fruits");
    var 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="images/strawberry.png">딸기
    <option value="images/banana.png" selected>바나나
    <option value="images/apple.png">사과
</select>

</form>
</body>
</html>
```



# 키 이벤트

43

- 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: 키 이벤트 리스너와 이벤트 객체의 프로퍼티

44

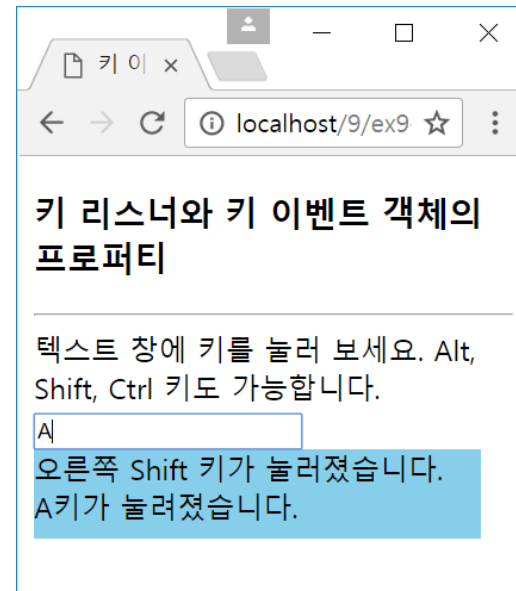
```
<!DOCTYPE html>
<html><head><title>키 이벤트</title>
<script>
function whatKeyDown(e) {
    var str = "";
    var div = document.getElementById("div");
    div.innerHTML = ""; // div 객체 내용을 지운다.
    if(e.altKey) {
        if(e.altLeft) str += "왼쪽 Alt 키가 눌러졌습니다.";
        else str += "오른쪽 Alt 키가 눌러졌습니다.";
        str += "<br>";
    }

    if(e.shiftKey) {
        if(e.shiftLeft) str += "왼쪽 Shift 키가 눌러졌습니다.";
        else str += "오른쪽 Shift 키가 눌러졌습니다.";
        str += "<br>";
    }

    if(e.ctrlKey) {
        if(e.ctrlLeft) str += "왼쪽 Ctrl 키가 눌러졌습니다.";
        else str += "오른쪽 Ctrl 키가 눌러졌습니다.";
        str += "<br>";
    }

    str += String.fromCharCode(e.keyCode) + "키가 눌러졌습니다."
    div.innerHTML = str; // div 객체에 문자열을 출력한다.
}
</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>
```



# onreset과 onsubmit

45

## □ onreset

- ▣ reset 버튼(<input type="reset">) 클릭 시
- ▣ false를 리턴하면 폼이 초기화되지 않음

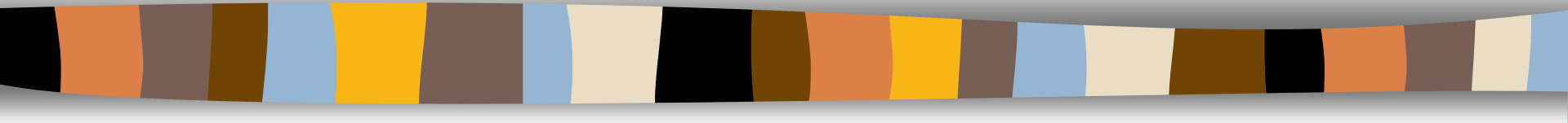
## □ onsubmit

- ▣ submit(<input type="submit">) 버튼 클릭 시
- ▣ false를 리턴하면 폼 전송하지 않음

## □ 리스너 작성

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

# Javascript BOM객체

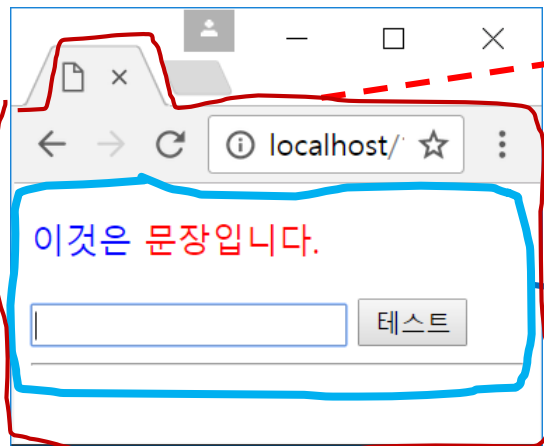


# 브라우저 관련 객체 개요

47

- BOM(Browser Object Model) 객체들
  - ▣ 자바스크립트로 브라우저를 제어하기 위해 지원되는 객체들
    - HTML 페이지의 내용과 관계없음
  - ▣ 브라우저 공통 BOM 객체들과 기능
    - window - 브라우저 윈도우 모양 제어. 새 윈도우 열기/닫기
    - navigator - 브라우저에 대한 다양한 정보 제공
    - history - 브라우저 윈도우에 로드한 URL 리스트의 히스토리 관리
    - location - 브라우저 윈도우에 로드된 HTML 페이지의 URL 관리
    - screen - 브라우저가 실행되고 있는 스크린 장치에 대한 정보 제공
- BOM의 국제 표준이 없다.
  - ▣ 브라우저마다 BOM 객체들이 조금씩 다름
  - ▣ 브라우저마다 이름이 같은 BOM 객체의 프로퍼티와 메소드 상이

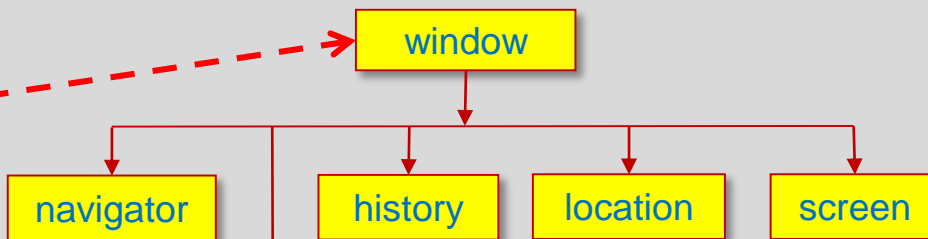
브라우저 관련 객체들



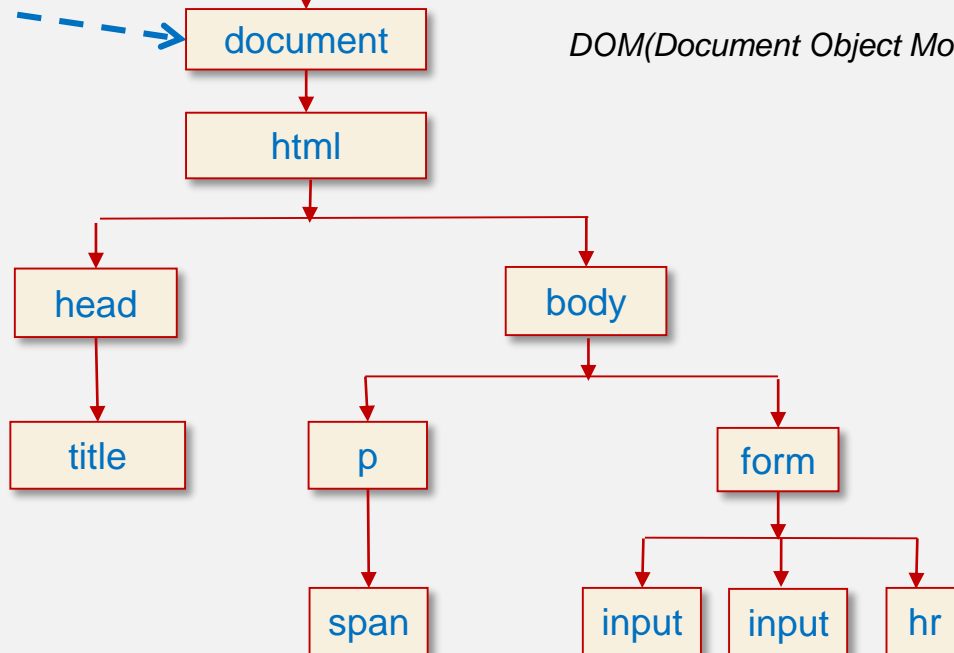
```
<!DOCTYPE html>
<html>
<head>
  <title>HTML DOM 트리</title>
</head>
<body>
<p style="color:blue" >이것은
  <span style="color:red">문장입니다.
</span>
</p>
<form>
  <input type="text" name="s">
  <input type="button" value="테스트">
  <hr>
</form>
</body>
</html>
```

HTML 문서의 내용과  
관련된 객체들

BOM(Browser Object Model)



DOM(Document Object Model)





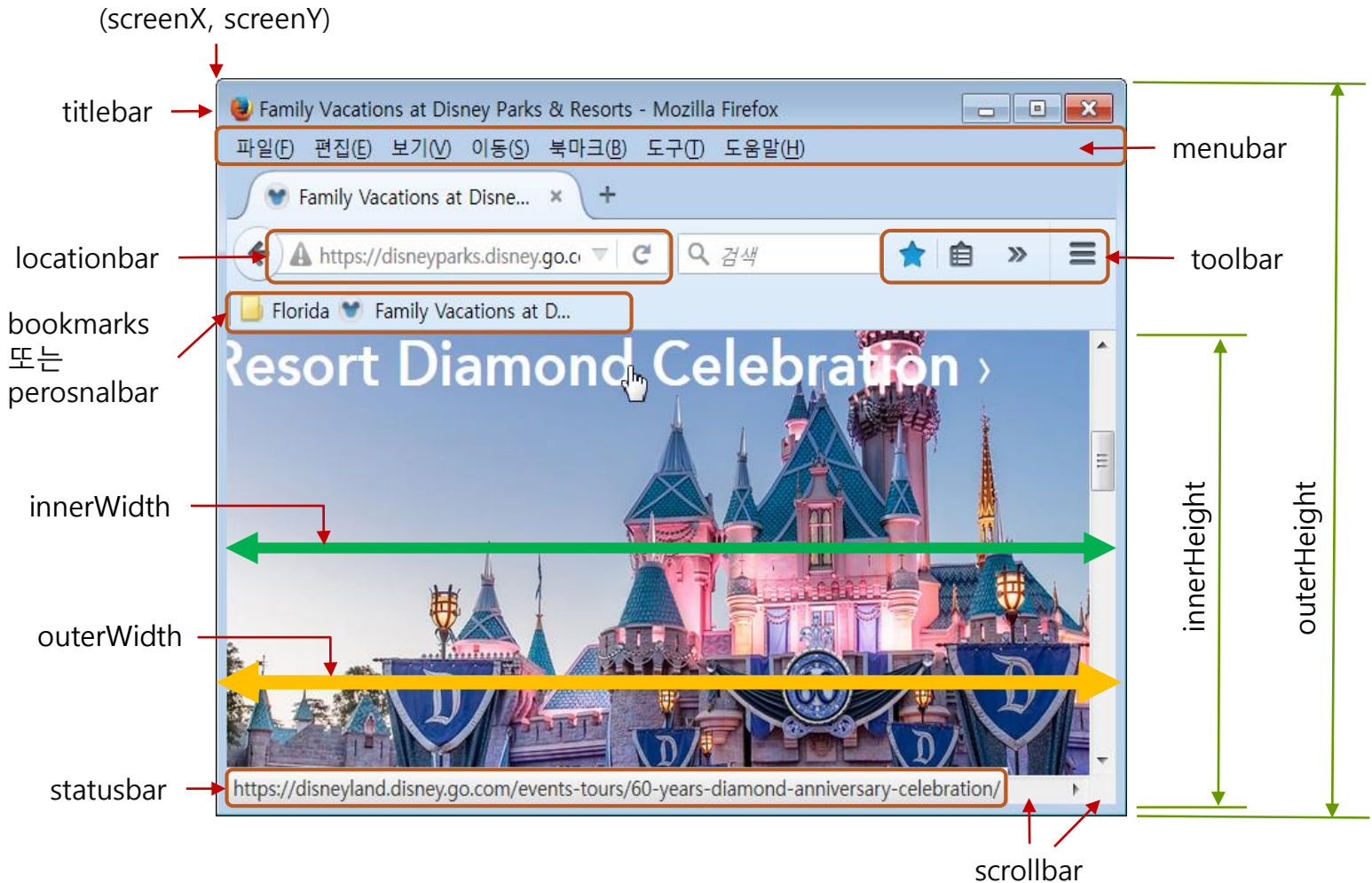
# window 객체

49

- window 객체
  - ▣ 열려 있는 브라우저 윈도우나 탭 윈도우의 속성을 나타내는 객체
  - ▣ 브라우저 윈도우나 탭 윈도우마다 별도의 window 객체 생성
- window 객체의 생성
  - ▣ 3 가지 경우
    - 브라우저가 새로운 웹 페이지를 로드할 때
    - <iframe> 태그 당 하나의 window 객체 생성
    - 자바스크립트 코드로 윈도우 열기 시 window 객체 생성
      - `window.open("웹페이지 URL", "윈도우이름", "윈도우속성")`,
- 자바스크립트 코드로 윈도우 객체에 대한 접근
  - ▣ window, 혹은 window.self, 혹은 self

# 윈도우 모양과 window 객체의 프로퍼티

50



# 윈도우 열기

51

## □ window.open()

### ▣ 윈도우를 새로 열고 웹 페이지 출력

■ 예)

```
window.open("http://www.naver.com", "", "");
```

### ▣ 3개의 매개변수를 가진 함수

```
window.open(sURL, sWindowName, sFeature)
```

- sURL : 윈도우에 출력할 웹 페이지 주소 문자열
- sWindowName : 새로 여는 윈도우의 이름 문자열로서 생략 가능
- sFeature : 윈도우의 모양, 크기 등의 속성들을 표현하는 문자열. 속성들은 빈칸 없이 콤마(‘,’)로 분리하여 작성하며 생략 가능

### ▣ 윈도우 이름(sWindowName)

_blank :	이름 없는 새 윈도우를 열고, 웹 페이지 로드
_parent :	현재 윈도우(혹은 프레임)의 부모 윈도우에 웹 페이지 로드
_self :	현재 윈도우에 웹 페이지 로드
_top :	브라우저 윈도우에 웹 페이지 로드

# 윈도우 열기 사례

52

- myWin 이름에 툴바만 가지는 새 윈도우 열고 sample.html 출력

```
window.open("sample.html", "myWin", "toolbar=yes");
```

- 현재 윈도우에 sample.html 출력

```
window.open("sample.html", "_self");
```

- 이름 없는 새 윈도우에 sample.html 출력

```
window.open("sample.html", "_blank");
```

- (10, 10) 위치에 300x400 크기의 새 윈도우 열고 네이버 페이지 출력

```
window.open("http://www.naver.com", "myWin",  
            "left=10,top=10,width=300,height=400");
```

- 이름과 속성이 없는 윈도우 열기

```
window.open("http://www.naver.com");  
window.open("http://www.naver.com", null, "");
```

- 빈 윈도우 생성

```
window.open();  
window.open("");
```

```
window.open("", "", "");  
window.open("", null, null);
```

# 윈도우 이름과 윈도우 열기

53

## □ 이름 없는 윈도우 열기

```
<button onclick="window.open('http://www.naver.com', '',  
    'width=600,height=600')">새 윈도우 열기  
</button>
```

- 버튼을 클릭할 때마다 새 윈도우를 열고 네이버 사이트 출력

## □ 이름을 가진 윈도우 열기

```
<button onclick="window.open('http://www.naver.com', 'myWin',  
    'width=600,height=600')">새 윈도우 열기  
</button>
```

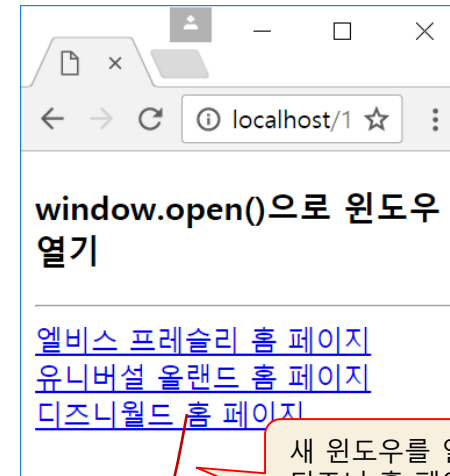
- myWin 이름의 윈도우가 열려 있지 않는 경우
  - 버튼을 클릭하면, myWin이름의 새 윈도우 열고 네이버 출력
- myWin 이름의 윈도우가 이미 열려 있는 경우
  - 버튼을 클릭하면, 이미 열려있는 myWin이름의 윈도우에 네이버 출력

# 예제: window.open()으로 윈도우 열기

54

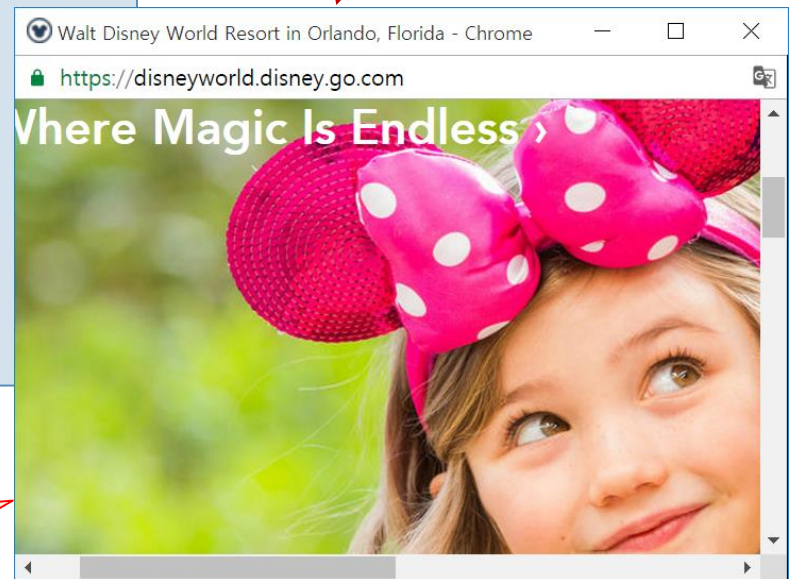
3개의 링크를 가진 웹 페이지를 작성하고, 각 링크를 클릭하면 myWin 이름의 새 윈도우를 열고 해당 사이트를 출력하라. myWin 윈도우는 공유된다. 새 윈도우는 스크린의 (300, 300) 위치에 400x300 크기로 출력된다.

```
<!DOCTYPE html>
<html>
<head>
<title>윈도우 열기</title>
<script>
function load(URL) {
    window.open(URL, "myWin", "left=300,top=300,width=400,height=300");
}
</script>
</head>
<body>
<h3>window.open()으로 윈도우 열기</h3>
<hr>
<a href="javascript:load('http://www.graceland.com')">
    엘비스 프레슬리 홈 페이지</a><br>
<a href="javascript:load('http://www.universallorlando.com')">
    유니버설 올랜드 홈 페이지</a><br>
<a href="javascript:load('http://www.disneyworld.com')">
    디즈니월드 홈 페이지</a><br>
</body>
</html>
```



새 윈도우를 열고  
디즈니 홈 페이지 출력

myWin 윈도우

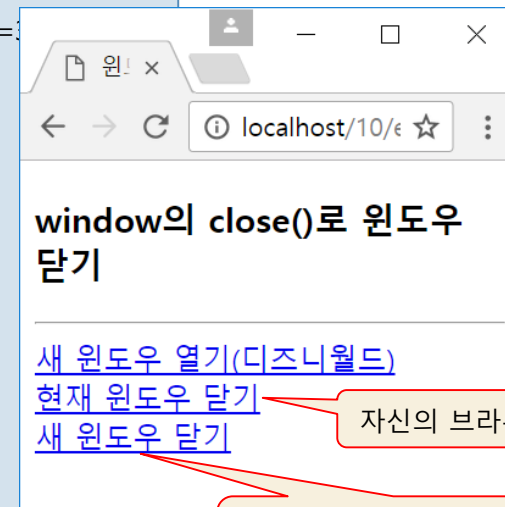


# 예제: 윈도우 닫기

55

윈도우를 스스로 닫는 경우와 자신이 생성한 윈도우를 닫는 사례를 보인다.

```
<!DOCTYPE html>
<html>
<head>
<title>윈도우 닫기</title>
<script>
var newWin=null; // 새로 연 윈도우 기억
function load(URL) {
    newWin = window.open(URL, "myWin", "left=300,top=300,width=400,height=300");
}
function closeNewWindow() {
    if(newWin==null || newWin.closed) // 윈도우가 열리지 않았거나 닫힌 경우
        return; // 윈도우가 없는 경우 그냥 리턴
    else
        newWin.close(); // 열어 놓은 윈도우 닫기
}
</script>
</head>
<body>
<h3>window의 close()로 윈도우 닫기</h3>
<hr>
<a href="javascript:load('http://www.disneyworld.com')">
    새 윈도우 열기(디즈니월드)</a> <br>
<a href="javascript:window.close()">
    현재 윈도우 닫기</a> <br>
<a href="javascript:closeNewWindow()">
    새 윈도우 닫기</a>
</body>
</html>
```



자신의 브라우저 윈도우 닫기

첫번째 링크에 의해 열려진 디즈니 월드 윈도우 닫기

# window 객체의 타이머 활용

56

- window 객체의 타이머 기능 2 가지
  - ▣ 타임아웃 코드 1회 호출
    - setTimeout()/clearTimeout() 메소드
  - ▣ 타임아웃 코드 반복 호출
    - setInterval()/clearInterval() 메소드



# setTimeout()/clearTimeout()

57

## □ setTimeout() : 타임아웃 코드 1회 실행

```
var timerID = setTimeout("timeOutCode", msec)
clearTimeout(timerID)
```

- timeOutCode : 타임아웃 자바스크립트 코드
- msec : 밀리초 단위의 정수로서, 타임아웃 지연 시간

setTimeout()은 msec 후에 timeOutCode를 1회 실행하도록 타이머를 설정하고, 타이머 ID를 리턴한다.  
clearTimeout()은 작동 중인 timerID의 타이머를 해제한다.

### 예) 3초 후 경고창 출력

```
function myAlert(time) {
    alert(time + "초 지났습니다");
}
var timerID = setTimeout("myAlert(3)", 3000); // 3초 후 myAlert('3') 호출
```

### 예) 3초가 되기 전에 타이머 해제

```
clearTimeout(timerID); // timerID의 타이머 해제
```

# 예제: setTimeout()로 웹 페이지 자동 연결

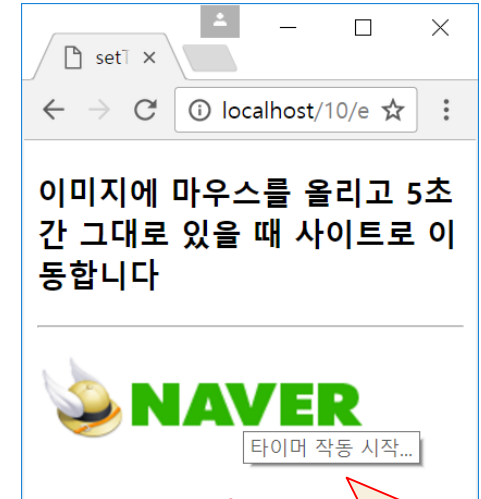
58

이미지 위에 마우스를 올린 상태로 5초가 지나면 네이버에 연결하며, 5초 전에 이미지를 벗어나면 타이머를 해제하는 코드를 작성하라.

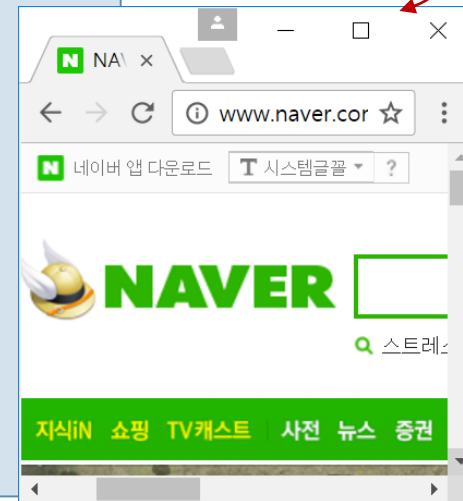
```
<!DOCTYPE html>
<html>
<head>
<title>setTimeout()으로 웹 페이지 자동 연결</title>
</head>
<body>
<h3>이미지에 마우스를 올리고 5초간 그대로 있을 때 사이트로 이동합니다</h3>
<hr>

<script>
var timerID=null;
function startTimer(time) {
  // 타이머 시작
  timerID = setTimeout("load('http://www.naver.com')", time);

  // 이미지에 마우스 올리면 나타내는 툴팁 메시지
  document.getElementById("img").title = "타이머 작동 시작...";
}
function cancelTimer() {
  if(timerID !=null)
    clearTimeout(timerID); // 타이머 중단
}
function load(url) {
  window.location = url; // 현재 윈도우에 url 사이트 로드
}
</script>
</body>
</html>
```



툴팁 메시지



마우스를 올리고  
5초간 그대로 있을 때

# setInterval()/clearInterval()

59

## □ setInterval() : 타임아웃 코드 반복 실행

```
var timerID = setInterval("timeOutCode", msec)
clearInterval(timerID)
```

- timeOutCode : 타임아웃 자바스크립트 코드
- msec : 밀리초 단위의 정수로서, 타임아웃 지연 시간

setInterval()은 msec 주기로 timeOutCode를 무한 반복하도록 타이머를 설정하고, 타이머의 ID를 리턴한다. clearInterval()은 timerID의 타이머를 해제한다.

예) 1초 간격으로 f() 반복 호출

```
function f() {
    // 함수 코드
}
var timerID = setInterval("f()", 1000); // 1초 주기로 f()가 호출되도록 타이머 작동
```

예) 타이머 해제

```
clearInterval(timerID); // timerID의 타이머 해제
```

# 예제: setInterval()로 텍스트 회전

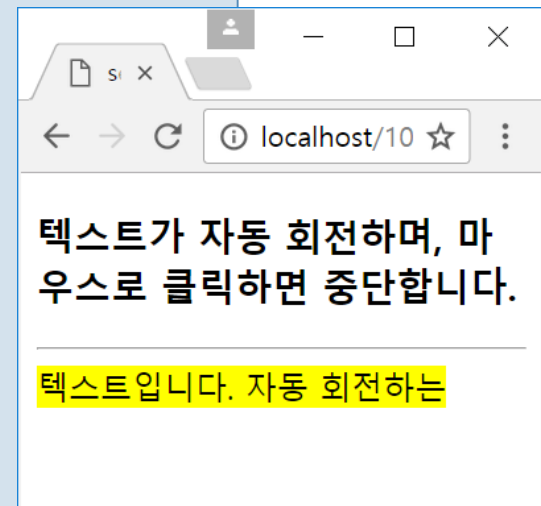
60

setInterval()을 이용하여 텍스트를 옆으로 반복 회전시키는 코드를 작성하라. 텍스트 위에 마우스를 클릭하면 회전이 중단된다.

```
<!DOCTYPE html>
<html>
<head> <title>setInterval()로 텍스트 회전</title> </head>
<body>
<h3>텍스트가 자동 회전하며, 마우스로 클릭하면 중단합니다.</h3>
<hr>
<div> <span id="div" style="background-color:yellow">
    자동 회전하는 텍스트입니다.</span>
</div>
<script>
var div = document.getElementById("div");
var timerID = setInterval("doRotate()", 200); // 200밀리초 주기로 doRotate() 호출

div.onclick = function (e) { // 마우스 클릭 이벤트 리스너
    clearInterval(timerID); // 타이머 해제. 문자열 회전 중단
}

function doRotate() {
    var str = div.innerHTML;
    var firstChar = str.substr(0, 1);
    var remains = str.substr(1, str.length-1);
    str = remains + firstChar;
    div.innerHTML = str;
}
</script>
</body> </html>
```



# 윈도우 위치 및 크기 조절

61

- 윈도우를 위로 5픽셀, 오른쪽으로 10픽셀 이동

```
window.moveBy(5, 10); 혹은  
moveBy(5, 10);
```

- 윈도우를 스크린의 (25, 10) 위치로 이동

```
window.moveTo(25, 10); 혹은 self.moveTo(25, 10);
```

- 윈도우 크기를 5 픽셀 좁게, 10픽셀 길게 조절

```
window.resizeBy(-5, 10); 혹은  
resizeTo(self.outerWidth-5, self.outerHeight+10);
```

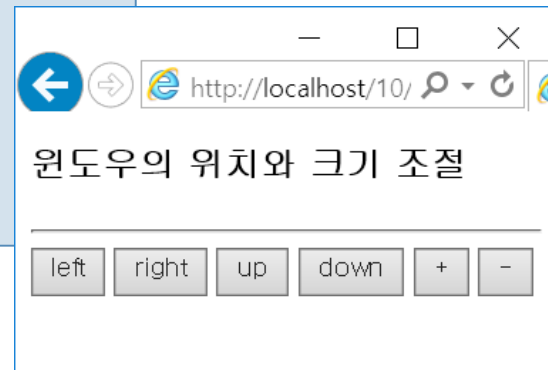
- 윈도우 크기를 200x300으로 조절

```
window.resizeTo(200, 300);
```

# 예제: 윈도우의 위치와 크기 조절

62

```
<!DOCTYPE html>
<html>
<head> <title> 윈도우의 위치와 크기 조절 </title> </head>
<body>
<h3> 윈도우의 위치와 크기 조절 </h3>
<hr>
<button onclick="window.moveBy(-10, 0)">left</button>
<button onclick="window.moveBy(10, 0)">right</button>
<button onclick="self.moveBy(0, -10)">up</button>
<button onclick="moveBy(0, 10)">down</button>
<button onclick="resizeBy(10, 10)">+</button>
<button onclick="resizeBy(-10, -10)">-</button>
</body>
</html>
```



\* 이 예제는 익스플로러에서는 잘 실행되지만, Chrome에서는 보안의 이유로 전혀 실행되지 않고, Edge에서는 크기 조절만 가능하다.

# 웹 페이지 스크롤

63

- 웹 페이지를 위로 10픽셀 스크롤(마우스 스크롤 다운)

```
window.scrollBy(0, 10); // 옆으로 0, 위로 10픽셀
```

- 웹 페이지를 왼쪽으로 10픽셀, 아래로 15픽셀 스크롤(마우스 스크롤 업)

```
window.scrollBy(10, -15);
```

- 웹 페이지의 (0, 200) 좌표 부분이 현재 윈도우의 왼쪽 상단 모서리에 출력되도록 스크롤

```
window.scrollTo(0, 200);
```

\* 스크롤 다운(scroll down)은 스크롤 바를 내리는 작동이며, 이에 따라 웹 페이지는 위로 이동한다.

# 예제: 1초마다 10픽셀씩 자동 스크롤

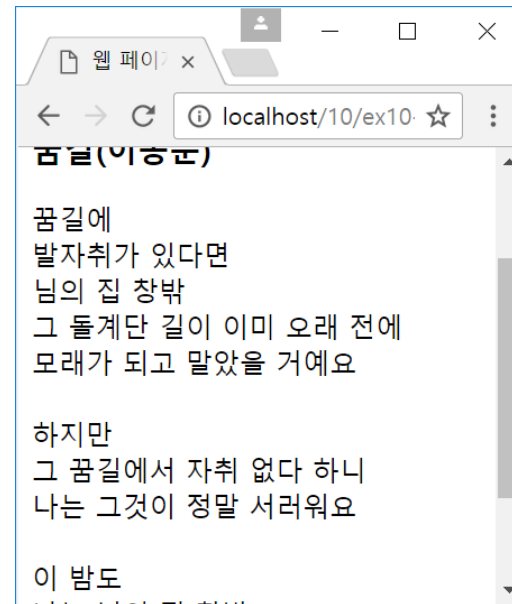
64

웹 페이지가 로드되자마자 자동으로 1초에 10픽셀씩 웹 페이지가 올라가도록 작성하라.

```
<!DOCTYPE html>
<html>
<head>
<title>웹 페이지의 자동 스크롤</title>
<script>
function startScroll(interval) {
    setInterval("autoScroll()", interval);
}

function autoScroll() {
    window.scrollBy(0,10); // 10픽셀 위로 이동
}
</script>
</head>
<body onload="startScroll(1000)">
<h3>자동 스크롤 페이지</h3>
<hr>
<h3>꿈길(이동순)</h3>
꿈길에<br>
발자취가 있다면<br>
님의 집 창밖<br>
그 돌계단 길이 이미 오래 전에<br>
모래가 되고 말았을 거예요<br><br>
하지만<br>
그 꿈길에서 자취 없다 하니<br>
나는 그것이 정말 서러워요<br><br>
이 밤도
```

```
그 꿈길에서 자취 없다 하니<br>
나는 그것이 정말 서러워요<br><br>
이 밤도<br>
나는 님의 집 창밖<br>
그 돌계단 위에 홀로 서서<br>
혹시라도 님의 목소리가 들려올까<br>
고개 숙이고 엿들어요<br>
</body>
</html>
```





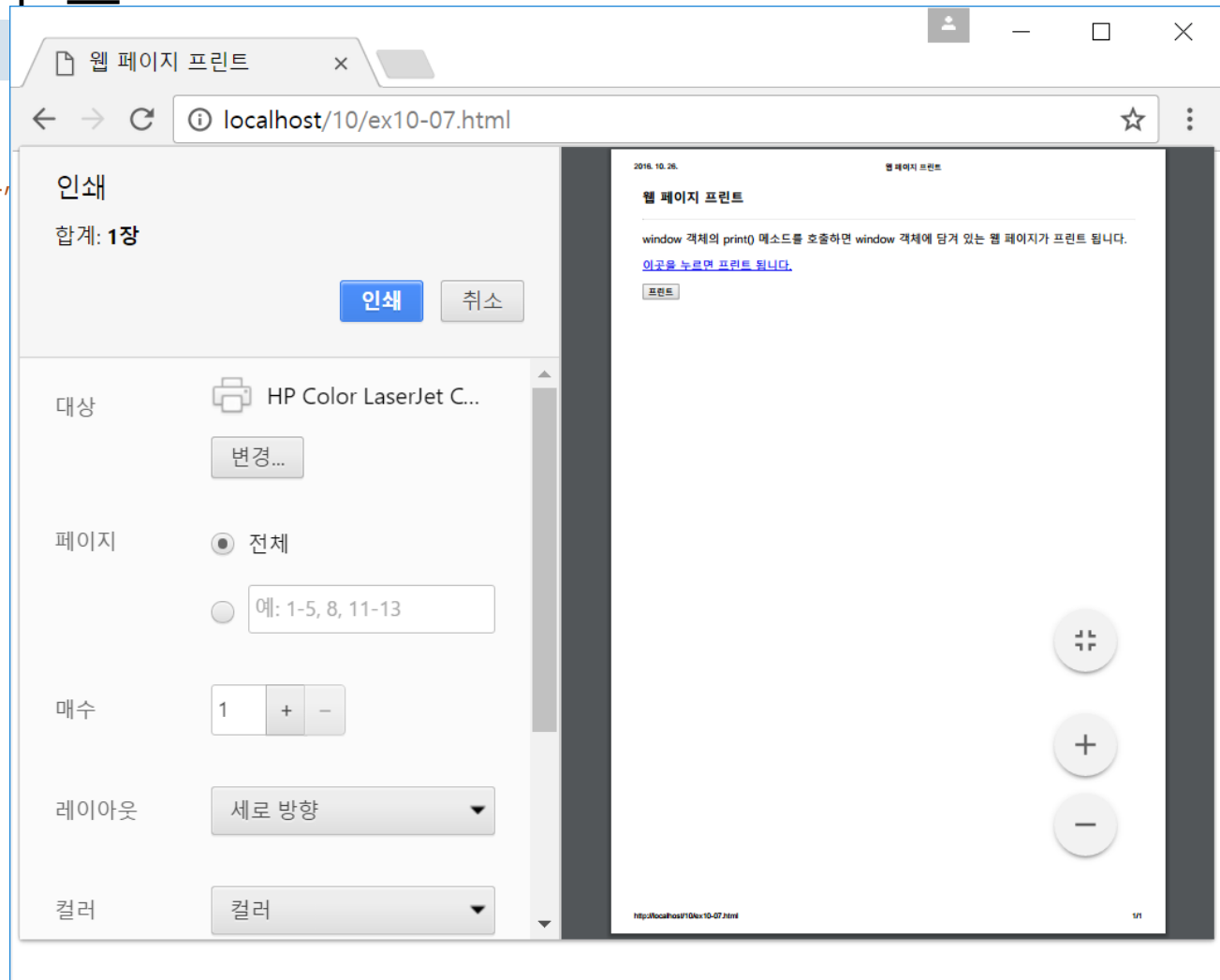
# 웹 페이지 프린트

65

## □ 웹 페이지 프린트

`window.print();`

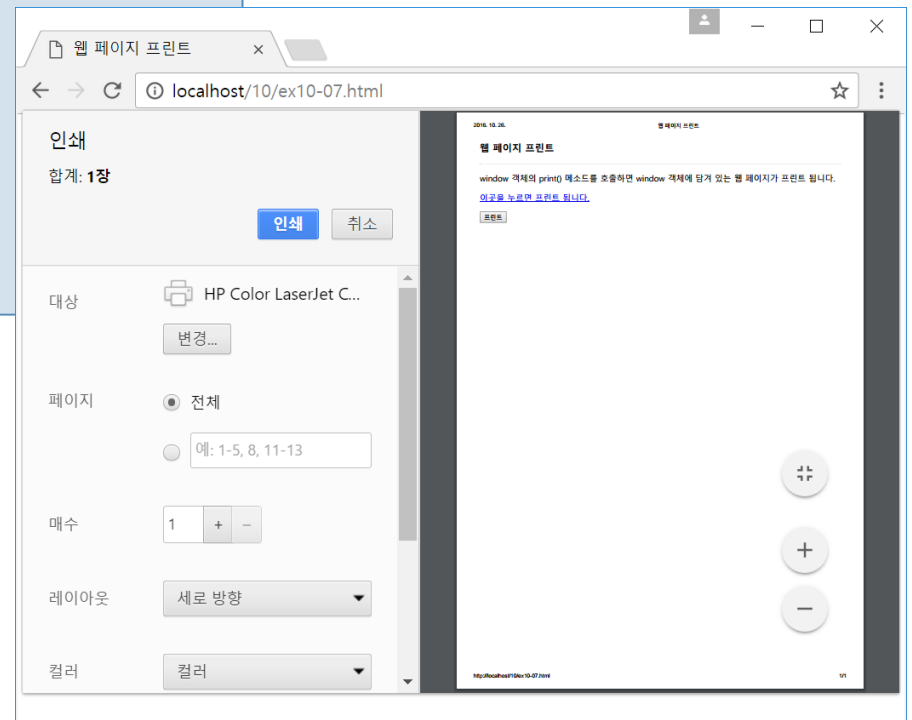
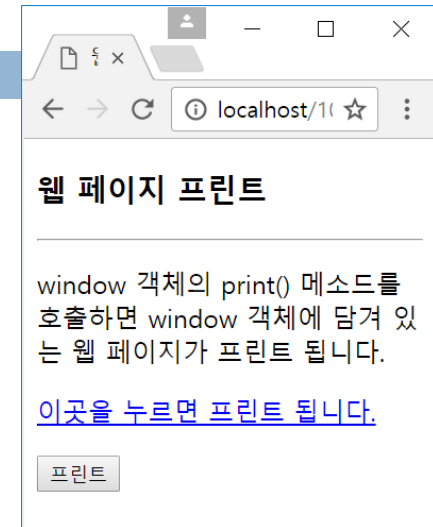
이 코드가 실행되면  
인쇄 다이얼로그가 열리고,  
'확인' 버튼을 누르면  
인쇄가 이루어진다.



# 예제: 웹 페이지 프린트

66

```
<!DOCTYPE html>
<html>
<head>
<title>웹 페이지 프린트</title> </head>
<body>
<h3>웹 페이지 프린트</h3>
<hr>
<p>window 객체의 print() 메소드를 호출하면
window 객체에 담겨 있는 웹 페이지가 프린트 됩니다.
<p>
<a href="javascript:window.print()">
  이곳을 누르면 프린트 됩니다.</a> <p>
<input type="button" value="프린트"
  onclick="window.print()">
</body>
</html>
```



# onbeforeprint와 onafterprint

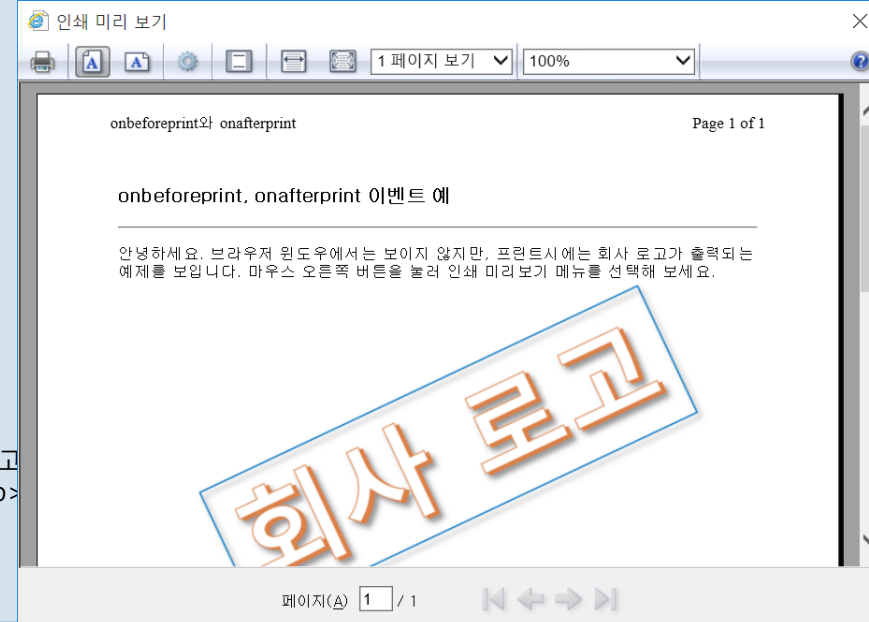
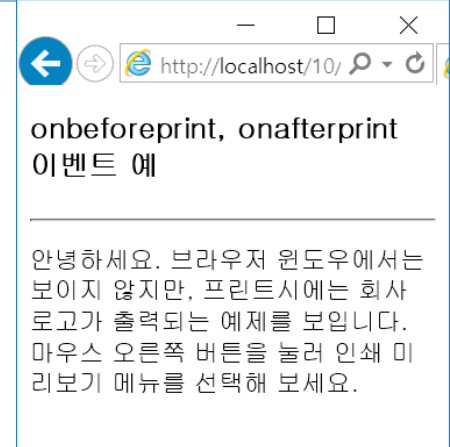
67

- 웹 페이지의 프린트 과정
  1. window 객체에 onbeforeprint 리스너 호출
  2. 웹 페이지 프린트
    - 브라우저가 웹 페이지를 이미지로 만들어 프린터로 전송
  3. window 객체에 onafterprint 리스너 호출
- onbeforeprint와 onafterprint 활용
  - ▣ 웹 페이지에는 보이지 않는 회사 로고를 프린트 시 종이에 출력
  - ▣ onbeforeprint
    - 회사 로그 이미지를 보이도록 CSS3 스타일 설정
  - ▣ onafterprint
    - 회사 로그 이미지를 보이지 않도록 CSS3 스타일 설정

# 예제: onbeforeprint와 onafterprint 이벤트 활용

68

```
<!DOCTYPE html>
<html>
<head> <title>onbeforeprint와 onafterprint</title>
<style>
#logoDiv {
    display : none;
    position : absolute; left : 0; top : 0;
    width : 100%; height : 100%;
}
</style>
<script>
window.onbeforeprint=function (e) {
    logoDiv = document.getElementById("logoDiv");
    logoDiv.style.display = "block"; // block으로 변경. 로고가 화면에 나타나게 함
}
window.onafterprint=hideLogo;
function hideLogo() {
    logoDiv = document.getElementById("logoDiv");
    logoDiv.style.display = "none"; // <div> 영역이 보이지 않게 함
    logoDiv.style.zIndex = -1; // 이미지를 문서의 맨 바닥으로 배치
}
</script> </head>
<body>
<h3>onbeforeprint, onafterprint 이벤트 예</h3>
<hr>
<div id="logoDiv">
    
</div>
<p>안녕하세요. 브라우저 윈도우에서는 보이지 않지만, 프린트시에는 회사 로고
보입니다. 마우스 오른쪽 버튼을 눌러 인쇄 미리보기 메뉴를 선택해 보세요.</p>
</body>
</html>
```



이 예제는 익스플로러와 Edge에서는 실행되지만, Chrome에서는 실행되지 않는다.

# location 객체

69

## □ location 객체

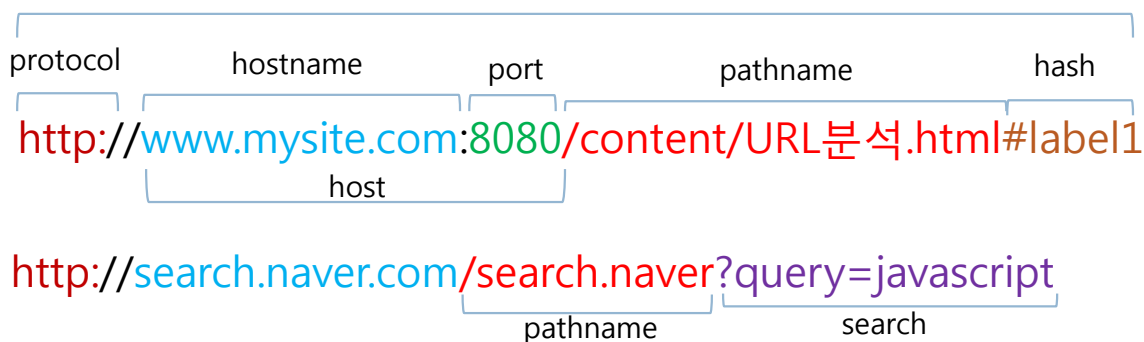
- 윈도우에 로드된 웹 페이지의 URL 정보를 나타내는 객체
- location 객체로 현재 윈도우에 웹 페이지 열기

```
window.location = "http://www.naver.com";  
window.location.href = "http://www.naver.com";  
window.location.assign("http://www.naver.com");  
window.location.replace("http://www.naver.com");
```

## □ 새 윈도우에 웹 페이지 열기

```
var win=window.open();           // 빈 윈도우 열기  
win.location="http://www.naver.com"; // 네이버 페이지 로드
```

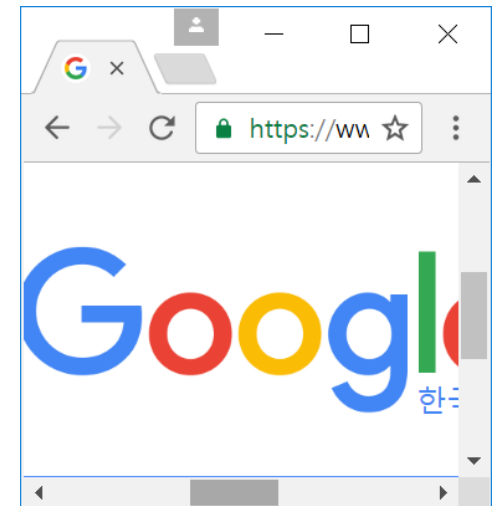
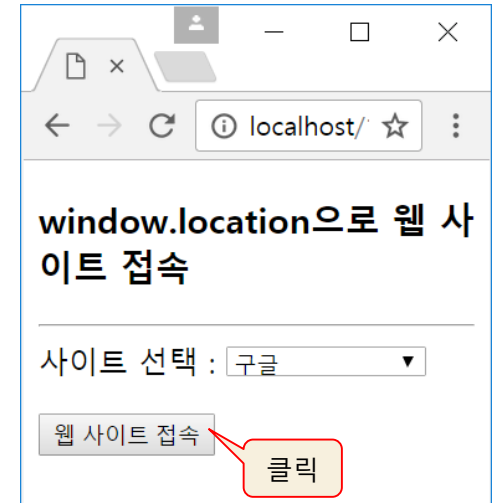
## □ location 객체의 프로퍼티와 URL의 구성 요소와의 관계



# 예제: location 객체로 웹 사이트 접속

70

```
<!DOCTYPE html>
<html>
<head> <title>window.location으로 웹 사이트 접속</title>
<script>
function load() {
    var select = document.getElementById("site");
    window.location=select.options[select.selectedIndex].value;
}
</script>
</head>
<body>
<h3>window.location으로 웹 사이트 접속</h3>
<hr>
사이트 선택 :
<select id="site">
    <option value="http://www.naver.com" selected> 네이버
    <option value="http://www.google.com"> 구글
    <option value="http://www.microsoft.com"> 마이크로소프트
</select>
<p>
<button onclick="load()"> 웹 사이트 접속 </button>
</body>
</html>
```



# navigator 객체

71

## □ navigator 객체

- ▣ 현재 작동중인 브라우저에 대한 다양한 정보를 나타내는 객체

프로퍼티	설명	r/w
appName	브라우저의 코드 이름을 가진 문자열	r
appVersion	브라우저 이름 문자열	r
platform	브라우저의 플랫폼과 버전에 관한 문자열	r
product	운영체제 플랫폼의 이름	r
userAgent	브라우저 엔진의 이름	r
vendor	브라우저가 웹 서버로 데이터를 전송할 때, HTTP 헤더 속의 user-agent 필드에 저장하는 문자열로서 웹 서버가 클라이언트를 인식하기 위한 목적	r
language	브라우저 제작 회사의 이름 문자열	r
onLine	브라우저의 언어를 나타내는 문자열로서, 영어는 "en-US", "ko-KR"	r
plugins	브라우저가 현재 온라인 작동중이면 true, 아니면 false	r
cookieEnabled	브라우저에 설치된 플러그인(plugin 객체)에 대한 컬렉션	r
geolocation	브라우저에 쿠키를 사용할 수 있는 상태이면 true, 아니면 false	r
	위치 정보를 제공하는 geolocation 객체에 대한 레퍼런스	r

# 예제: navigator로 브라우저 정보 출력

72

```
<!DOCTYPE html>
<html>
<head><title>브라우저 정보 출력</title>
<style>
span { color : red; }
div {
    border-color : yellowgreen;
    border-style : solid;
    padding : 5px;
}
</style>
<script>
function printNavigator() {
    var text = "<span>appName</span>: " + navigator.appCodeName + "<br>";
    text += "<span>appName</span>: " + navigator.appName + "<br>";
    text += "<span>appVersion</span>: " + navigator.appVersion + "<br>";
    text += "<span>platform</span>: " + navigator.platform + "<br>";
    text += "<span>product</span>: " + navigator.product + "<br>";
    text += "<span>userAgent</span>: " + navigator.userAgent + "<br>";
    text += "<span>vendor</span>: " + navigator.vendor + "<br>";
    text += "<span>language</span>: " + navigator.language + "<br>";
    text += "<span>onLine</span>: " + navigator.onLine + "<br>";
    text += "<span>cookieEnabled</span>: " + navigator.cookieEnabled + "<br>";
    text += "<span>javaEnabled</span>: " + navigator.javaEnabled() + "<br>";
    text += "<span>plugins.length</span>: " + navigator.plugins.length + "<br>";
    for(j=0; j<navigator.plugins.length; j++) {
        text += "plugins" + j + " : <blockquote>";
        text += navigator.plugins[j].name + "<br>";
        text += "<i>" + navigator.plugins[j].description + "</i> <br>";
        text += navigator.plugins[j].filename + "</blockquote>";
    }

    // div 태그에 출력
    var div = document.getElementById("div");
    div.innerHTML = text;
}
```

```
</script>
</head>
<body onload="printNavigator()">
<h3>브라우저에 관한 정보 출력</h3>
아래에 이 브라우저에 관한 여러 정보를 출력합니다.
<hr>
<p>
<div id="div"></div>
</body>
</html>
```



브라우저 정보 출력

localhost/10/ex10-10.html

## 브라우저에 관한 정보 출력

아래에 이 브라우저에 관한 여러 정보를 출력합니다.

```
appCodeName: Mozilla
appName: Netscape
appVersion: 5.0 (Windows NT 10.0; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/54.0.2840.99 Safari/537.36
platform: Win32
product: Gecko
userAgent: Mozilla/5.0 (Windows NT 10.0; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/54.0.2840.99 Safari/537.36
vendor: Google Inc.
language: ko
onLine: true
cookieEnabled: true
javaEnabled():false
plugins.length: 5
plugins0 :
    Widevine Content Decryption Module
    Enables Widevine licenses for playback of
    HTML audio/video content. (version:
    1.4.8.903)
    widevinecdmadapter.dll
plugins1 :
    Shockwave Flash
    Shockwave Flash 23.0 r0
```

플러그인 이름

플러그인 설명

플러그인 파일

# screen 객체

74

## □ screen

- ▣ 브라우저가 실행되는 스크린 장치에 관한 정보를 담고 있는 객체

프로퍼티	설명	r/w
availHeight	작업 표시줄 등을 제외하고 브라우저가 출력 가능한 영역의 높이	r
availWidth	작업 표시줄 등을 제외하고 브라우저가 출력 가능한 영역의 폭	r
pixelDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수	r
colorDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수로서 pixelDepth와 동일. 대부분의 브라우저에서 지원되므로 pixelDepth보다 colorDepth를 사용할 것을 권함	r
height	스크린의 수직 픽셀 수	r
width	스크린의 수평 픽셀 수	r

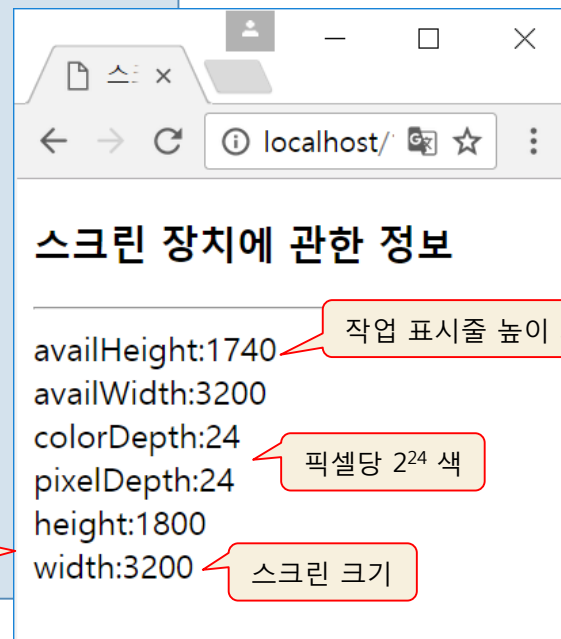
# 예제: 스크린 장치에 관한 정보 출력

75

```
<!DOCTYPE html>
<html>
<head>
<title>스크린 장치에 관한 정보 출력</title>
<script>
function printScreen() {
    var text = "availHeight:".fontcolor('blue') + screen.availHeight + "<br>";
    text += "availWidth:".fontcolor('blue') + screen.availWidth + "<br>";
    text += "colorDepth:".fontcolor('blue') + screen.colorDepth + "<br>";
    text += "pixelDepth:".fontcolor('blue') + screen.pixelDepth + "<br>";
    text += "height:".fontcolor('blue') + screen.height + "<br>";
    text += "width:".fontcolor('blue') + screen.width + "<br>";

    document.getElementById("div").innerHTML = text;
}
</script>
</head>
<body onload="printScreen()">
<h3>스크린 장치에 관한 정보</h3>
<hr>
<div id="div"></div>
</body>
</html>
```

height와 width는 브라우저의 설정에서 확대/축소 값을 100%로 해야 정확한 값으로 출력됨



# history 객체

76

## □ history 객체

- 윈도우에서 방문한 웹 페이지 리스트(히스토리)를 나타내는 객체

프로퍼티	설명	r/w
length	히스토리 리스트에 있는 엔트리 수	r

메소드	설명
back()	히스토리에 있는 이전 웹 페이지로 이동. 브라우저의 <back> 버튼과 동일
forward()	히스토리에 있는 다음 웹 페이지로 이동. 브라우저의 <forward> 버튼과 동일
go(n)	히스토리에서 현재 웹 페이지에서 n 만큼 상대적인 웹 페이지로 이동

- history 객체를 이용하여 웹 페이지를 이동하는 코드 사례

```
history.back();    // 이전 페이지로 이동
history.go(-1);    // 이전 페이지로 이동
history.forward(); // 다음 페이지로 이동
history.go(1);     // 다음 페이지로 이동
```

# 예제: history 객체 활용

77

```
<!DOCTYPE html>
<html>
<head> <title>history 객체 활용</title> </head>
<body>
<h3>history 객체 활용</h3>
<hr>
<button onclick="history.back()">back()</button>
<button onclick="history.forward()">forward()</button>
<button onclick="history.go(-1)">go(-1)</button>
</body>
</html>
```

