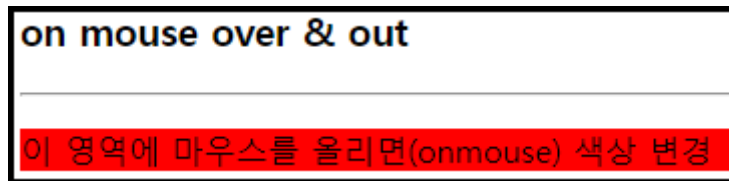


1] onmouseover & onmouseout : 마우스 액션 스크립트

- onmouseover : 마우스의 커서가 특정 영역에 도달시 적용할 기능
- onmouseout : 마우스의 커서가 특정 영역에서 벗어날시 적용할 기능
 - HTML의 <head><script> 영역 혹은 <body> 영역에서만 작성

☐ 구현 목표 : 아래의 이미지



마우스를 올리면 *red*, 벗어나면 *white*를 적용

- ☐ 오늘의 코드는 자바스크립트로 옮기는 것이 좋으나, HTML에서만 코드를 작성하면 한 화면에서 기능을 이해하기 편리하기 때문
- ☐ 가급적 function a,b,c 로 함수명을 통일한다. 이는 내장함수와 구분하기 위함
- ☐ 금일의 학습과정에서는 5개 정도의 코드로 위 이미지와 동일한 결과를 구현
- ☐ 5개의 기능을 학습하는데 중점, 5개중 하나를 구현가능해도 무방

1) onmouseover & onmouseout 코드의 직접 작성

```
<body>
  <h3>on mouse over & out</h3><hr>
  <p onmouseover="this.style.backgroundColor='red'"
    onmouseout="this.style.backgroundColor='white'"
  >이 영역에 마우스를 올리면(onmouse) 색상 변경</p>
</body>
```

- 5개 중 유일하게 <body> 에서 작성을 종료
- tag의 내용이 길어지는 단점이 있다...고는 하나 단순 길이로는 가장 짧은 편
- BOM, DOM이 문제?, 이유는 설명이 부족

- ❖ 다른 4가지 방법은 공통적으로 <body> 에 아래의 코드를 작성
- ❖ <head> 의 <script> 영역에서의 코드만 다소 차이가 있는 편

```
<body onload="init()">
  <h3>리스너 작성</h3><hr>
  <p id="p"> 마우스를 올리면(onmouse) 색상 변경</p>
</body>
```

2) 각각의 function 을 지정

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <script>
      function a(){
        p=document.getElementById("p");
        p.onmouseover = b;
        p.onmouseout = c;
        function b(){p.style.backgroundColor="red";}
        function c(){p.style.backgroundColor="white";}
      }
    </script>
  </head>

  <body onload="a()">
    <h3>리스너 작성</h3><hr>
    <p id="p"> 이 영역에 마우스를 올리면(onmouse) 색상 변경</p>
  </body>
</html>
```

전체 화면, 이후의 설명에서는 `<body>` 부분이 중복이므로 생략하고, `head`의 `script` 부분만 작성하겠음

- `let p` : 변수의 선언, 없어도 자바스크립트는 허용, 본문에 `p`가 많이 사용되므로 편의상 선언함
- `function init() : init()`이라는 내장함수는 없음¹, 편의상 `a`로 대체
- `document.getElementById("p")`
 - `document` : 현재 (HTML) 문서 '의'
 - `getElementById("p")` : `id="p"`가 지정된 곳의 내용을 가져오겠다.
 - `getElement` : 해당되는 요소를 가져오겠다
 - `ById` : `id`로 지정된 곳에서
 - `("p")` : 본문에서는 `id="p"`를 의미, `p`가 위치한 곳에 기능을 적용하기 위함
- `p.onmouseover = b` : `p`의 영역에 마우스가 올라오면 `b` 함수의 내용을 적용,
`p.onmouseout = c` : `p`의 영역에 마우스가 올라오면 `c` 함수의 내용을 적용
 - `function (b)` : 마우스를 올리면 `red` 색상 적용
 - `function (c)` : 벗어나면 `white` 색상 적용
- `<p id="p">` : 해당 태그로 인하여 `<p>` 함수의 영역과 함수 `b, c`의 색상 기능을 연결
 - `<p>` 함수의 영역 : 브라우저의 "마우스를 올리면(onmouse) 색상 변경"을 의미
- `onload`² : 문서의 모든 콘텐츠(images, script, css, etc)가 로드된 후 발생하는 이벤트
 1. `<head><script>` 부분을 임시로 읽고
 2. `<body>`의 내용을 포함한 모든 콘텐츠를 로드후
 3. `<onload>` 부분을 시행

¹ 객체 초기자 : `new object()`와 `object.create()`를 사용 - Mdn web docs

² [Javascript] [onload, ready](#)

3) addEventListener 를 활용한 지정

```
<script>
  let p;
  function a(){
    p=document.getElementById("p");
    p.addEventListener("mouseover", b); //p.onmouseover = b; 와 동일
    p.addEventListener("mouseout", c); //p.onmouseout = c; 와 동일
    function b(){p.style.backgroundColor="red";}
    function c(){p.style.backgroundColor="white";}
  }
</script>
```

- .addEventListener("mouseover", b)
- .addEventListener('이벤트타입', 실행할 함수)
- .addEventListener() : (괄호안의 내용)을 듣고(listener) 처리(add)하겠다
 - 이벤트가 발생했을때 그 처리를 담당하는 함수로 이벤트 핸들러라고도 함
 - 지정된 타입의 이벤트가 특정 요소에서 발생하면, 웹 브라우저는 그 요소에 등록된 이벤트 리스너를 실행
- mouseover : 이벤트타입, '마우스를 올리면' 을 의미
- b : 실행할 함수명
- .addEventListener는 소스코드가 많아질 경우 사용하는 것이 권장

4) 참조변수 this

```
<script>
  let p;
  function a(){
    p=document.getElementById("p");
    p.onmouseover=function(){this.style.backgroundColor="red"};
    p.onmouseout=function(){this.style.backgroundColor="white"};
  }
</script>
```

- 참조변수 this : 자기자신을 포함한 가장 가까운 객체를 호출(현재 a 를 의미)
- 앞서 function b,c 를 선언하지 않고 비교적 간략히 구성됨

5) EventListener + 참조변수 this

```
<script>
  let p;
  function a(){
    p=document.getElementById("p");
    p.addEventListener("mouseover", function(){this.style.backgroundColor="red"});
    p.addEventListener("mouseout", function(){this.style.backgroundColor="white"});
  }
</script>
```

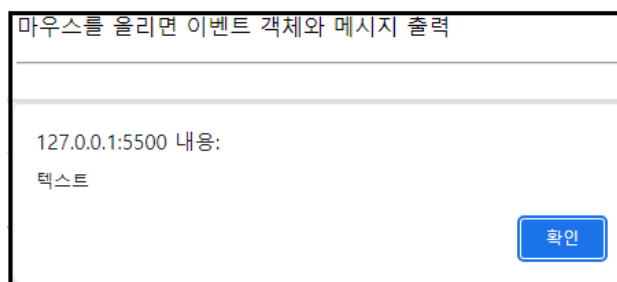
- .addEventListener('이벤트타입', 실행할 함수)
 - 이벤트 타입 : mouseover 가 적용
 - 실행할 함수 : 참조변수 this를 활용

2] Alert, Confirm : 알림 스크립트

- ☐ 알림을 나타내는 스크립트는 Alert, Confirm, Prompt 가 존재
- ☐ 이중 Alert와 Confirm 에 대해 학습

1) alert : event가 발생시 이벤트 객체(메시지를 담은 작은 이벤트 창)가 등장

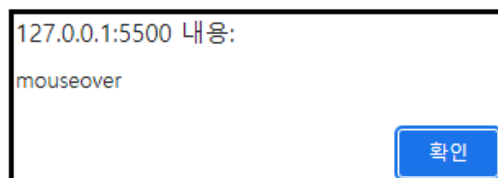
```
<body>
  <p id="p">마우스를 올리면 이벤트 객체와 메시지 출력</p><hr>
  <script>
    function a(){alert("텍스트");}
    document.getElementById("p").onmouseover= a;
  </script>
</body>
```



메시지("텍스트")가 담긴 이벤트 창이 열린다

- document.getElementById("p").onmouseover=a;
 - document.getElementById("p") : 본 문서의 p 영역에
 - .onmouseover = 마우스를 올리면
 - =a : a 함수를 실행, 이때 a 함수는 alert창과 텍스트가 출력
- function a(){alert("텍스트");}
 - 함수 a 를 호출시 alert 창이 뜨고 해당 창에 텍스트가 출력
- function a(b){alert("b.type");}

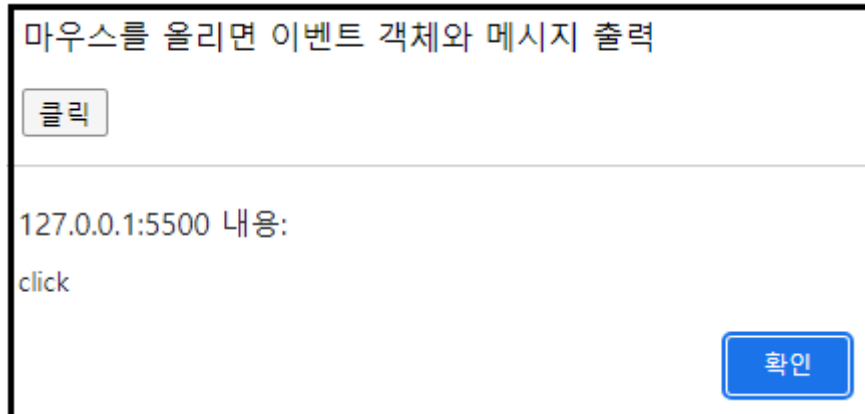
```
<body>
  <p id="p">마우스를 올리면 이벤트 객체와 메시지 출력</p>
  <script>
    function a(b){alert(b.type);}
    document.getElementById("p").onmouseover=a;
  </script>
</body>
```



- 이전에는 "텍스트"가 출력되었지만 현재는 mouseover 가 뜬
- function a(b){alert(b.type);}
 - 파라미터 (b)와 b.type가 연결,
 - type : 현재 발생하는 이벤트가 어떤 것인지 구분
 - mouseover에 의한 것임을 출력

- 버튼을 이용한 alert : 버튼(클릭)을 누르면 이벤트창과 type 로 인한 click 메시지 출력

```
<body>
  <p id="p">마우스를 올리면 이벤트 객체와 메시지 출력</p>
  <button onclick="a(event)">클릭</button>
  <script>
    function a(b){alert(b.type);}
    document.getElementById("p").onmouseover=a;
  </script>
</body>
```



- onclick="a(event)" : 사용자가 클릭시 a 함수의 기능이 담긴 이벤트가 발생
 - onclick : 브라우저에서 사용자가 클릭시
 - event : 이벤트가 발생, 일반적인 매개변수가 아닌 내장함수로서의 event³
 - a(event) : a 함수의 이벤트를 발생시켜라

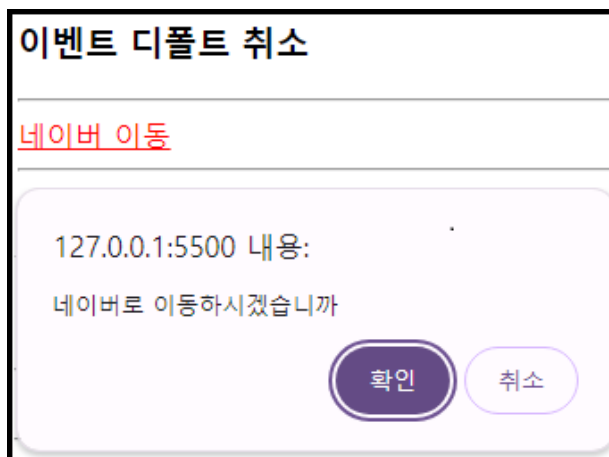
```
<body>
  <p id="p">마우스를 올리면 이벤트 객체와 메시지 출력</p>
  <button onclick="f(event)">클릭</button>
  <script>
    function f(e){alert(e.type);}
    document.getElementById("p").onmouseover=f;
  </script>
</body>
```

- function f(e) : a,b,c 로 해도 상관은 없으나 f(event)와 연결된 함수란 의미로 e 를 사용하는 것이 관례

³ [JavaScript - 이벤트\(Event\), 이벤트의 종류, 이벤트 연결](#)

- 2) confirm : event가 발생시 이벤트 객체(메시지를 담은 작은 이벤트 창)이 등장,
유저의 선택에 따라 확인(True) 와 취소(False) 의 연결값을 실행

```
<head>
  <meta charset="UTF-8" />
  <script>
    function a(){confirm("네이버로 이동하시겠습니까");}
    function noAction(e){e.preventDefault();}
  </script>
</head>
<body>
  <h3>이벤트 디폴트 취소</h3><hr>
  <a href="https://www.naver.com" onclick="return a()">네이버 이동</a><hr>
</body>
```



- confirm("텍스트") : 알림 스크립트, 텍스트의 내용을 알림창에 출력
- function noAction(e){e.preventDefault();} : 취소(False)시 취소(False)가 됨, 해당 코드가 없으면 취소(False)를 눌러도 확인(True)로 인식
 - noAction :
 - preventDefault : 이벤트의 디폴트 행동을 취소
 - 구문 전체를 그대로 쓰는 것을 권장

```
<head>
  <meta charset="UTF-8" />
  <title>이벤트 디폴트 취소</title>
  <script>
    function query(){let ret = confirm("네이버 이동"); return ret;}
    function noAction(e){e.preventDefault();}
  </script>
</head>
<body>
  <h3>이벤트 디폴트 취소</h3><hr>
  <a href="https://www.naver.com" onclick="return query()">네이버 이동</a><hr>
</body>
```

강사님이 적은 원본, query() 도 관례일까?

- 클릭시 이벤트 객체의 정보를 출력

```
<body>
  <p id="p">클릭하면 이벤트 객체 출력</p>
  <button onclick="f(event)">클릭</button>
  <script>
    function f(e){
      let text = "type : " + e.type + "<br>"
        + "target : " + e.target + "<br>"
        + "currentTarget : " + e.currentTarget + "<br>"
        + "preventDefault : " + e.preventDefault + "<br>"
        + "defaultPrevented : " + e.defaultPrevented;
      let p=document.getElementById("p");
      p.innerHTML = text;
    }
  </script>
</body>
```

```
type : click
target : [object HTMLButtonElement]
currentTarget : [object HTMLButtonElement]
preventDefault : function preventDefault() { [native code] }
defaultPrevented : false
```

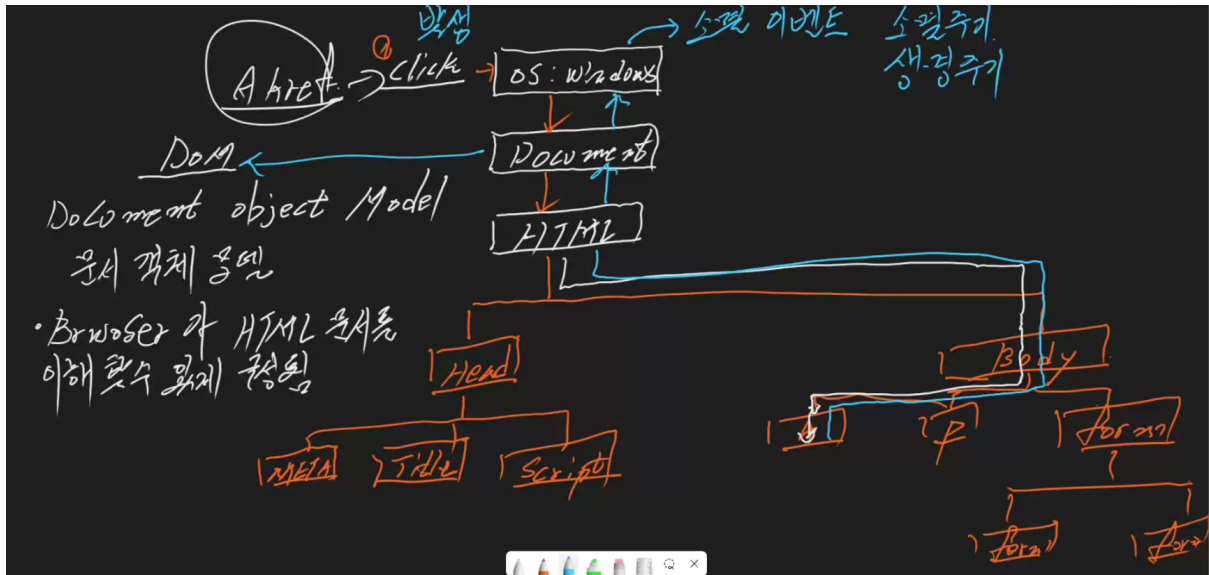
클릭

- type : 이벤트 종류, click, load
- target : 이벤트 발생 객체
- currentTarget : DOM
- preventDefault : 디폴트 초기값 취소
- defaultPrevented : 이벤트 취소시의 상태

3] DOM & BOM

 보충 필요

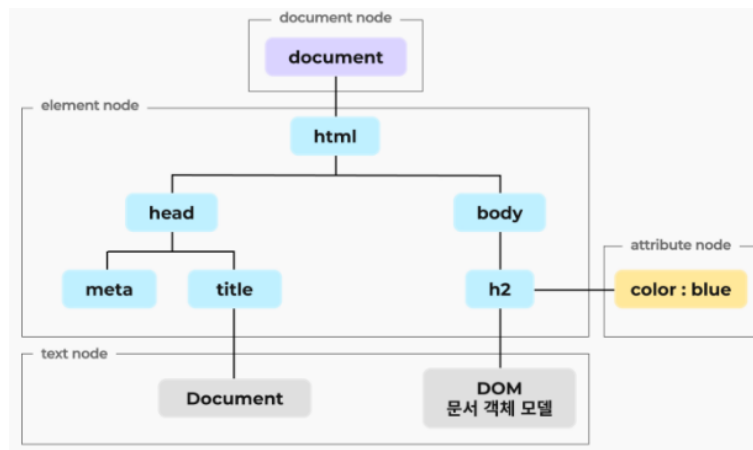
1) DOM(Documents Object Model) : 브라우저가 HTML 문서를 이해할 수 있게 구성⁴



- 윈도우os에서 클릭하여 인풋박스가 생성
 - 웹브라우저가 작동할때 Document를 생성
 - 최초의 문서를 생성하는 것을 DOM
- BOM(Browser Object Model) : 자바스크립트가 주체로 브라우저와의 소통이 목적
 - DOM 생성이후 BOM 생성
 - 자바스크립트의 명령어를 시행하지 말라고?
 - BOM 이 존재하기에 SPA가 가능, 동적처리 가능함을 의미
 - windows : 각 프레임별 하나씩 존재(object)
 - location : URL 정보
 - documents : 현재 문서 정보
 - navigator : 브라우저간 호환성
 - history : 방문 기록
 - screen : 브라우저 외부 환경

- DOM 보충 필요⁵

- 정의 : 웹 페이지(HTML이나 XML 문서)의 콘텐츠 및 구조, 그리고 스타일 요소를 구조화시켜 표현하여 프로그래밍 언어가 해당 문서에 접근하여 읽고 조작할 수 있도록 API를 제공하는 일종의 인터페이스
- 필요성 : 정적인 웹이라면 HTML, CSS로도 충분하겠지만 그 이상의 인터랙티브한 기능을 구현하고자 한다면 자바스크립트와 DOM이 필요
- 역할 : 자바스크립트 같은 스크립팅 언어가 쉽게 웹 페이지에 접근하여 조작할 수 있게끔 연결
- 구조 : 웹 페이지, 즉 HTML 문서를 계층적 구조와 정보로 표현하며, 이를 제어할 수 있는 프로퍼티와 메서드를 제공하는 트리 자료구조, HTML DOM, 혹은 HTML DOM Tree로 부르기도 함



- 코딩 파일명 작성법

- **camel** : 첫 문자의 첫 글자는 소문자, 연결된 뒷문자의 첫 글자는 대문자 #myTeam
 - 시작하는 위치를 알 수 있어서 보편적
- **kebab** : 모든 문자는 소문자, 연결될 문자는 -(hyphen)으로 연결 #my-team
- **snake** : 모든 문자는 소문자, 연결될 문자는 _(underbar)으로 연결 #my_team
- **pascal** : 모든 문자의 첫 글자는 대문자 #MyTeam

=====아래부터 미정리=====

⁵ [문서 객체 모델 DOM 과 자바스크립트 JavaScript | 생성 방식 및 접근 방법](#) - codestate

4] Capture & Bubble

```
<body>
  <h3>문장 확인</h3><hr>
  <p style="color : ■aquamarine">이건<span style="color : ■red">문장이다</span></p>
  <form>
    <input type="text">
    <input type="button" value="test" id="button">
  </form>
  <div id="div" style="color : ■green"></div>
  <script>
    let div = document.getElementById("div");
    let button = document.getElementById("button");

    document.body.addEventListener("click", capture, true);
    button.addEventListener("click", bubble, false);
    document.body.addEventListener("click", bubble, false);

    function capture(e){
      let obj = e.currentTarget;
      let tagName = obj.tagName;
      div.innerHTML += "<br> capture 단계 : " + tagName + "태그" + e.type + "이벤트"
    }
    function bubble(e){
      let obj = e.currentTarget;
      let tagName = obj.tagName;
      div.innerHTML += "<br> bubble 단계 : " + tagName + "태그" + e.type + "이벤트"
    }
  </script>
</body>
</html>
```

문장 확인

이건문장이다

capture 단계 : BODY태그click이벤트
bubble 단계 : INPUT태그click이벤트
bubble 단계 : BODY태그click이벤트

- `let div = document.getElementById("div")` : 이벤트 메시지 출력 공간, div 실행 공간
- `let obj=e.currentTarget` : 이벤트 발생 DOM 안에 기록, 현재 이벤트를 받은 DOM 객체
- `div.innerHTML +=` : 추가, 연속해서 써내려감
- `capture` 단계 : 이벤트를 발생시킨 객체를 찾아 내려가는 단계
- `bubble` 단계 : 이벤트를 발생시킨 객체를 찾고 되돌아가는 단계
- `currentTarget`

- `eval()` : 계산의 결과값을 반환

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>calculate</title>
    <script>
      function calculate(){
        let exp = document.getElementById("exp");
        let result = document.getElementById("result");
        result.value = eval(exp.value);
      }
    </script>
  </head>
  <body>
    <h3></h3><hr>
    계산수식을 입력
    <br><br>
    <form>
      식 : <input type="text" id="exp" value=""><br>
      값 : <input type="text" id="result" value=""><br>
      <input type="button" value="계산" onclick="calculate()">
    </form>
  </body>
</html>
```

계산수식을 입력

식 :

값 :

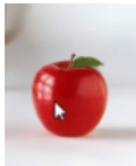
- 자바스크립트의 `eval()` 함수는 문자열로 이루어진 스크립트 코드를 실행하는 함수입니다.
- `eval()`은 매개변수로 받은 문자 값을 값으로 사용하는 게 아니라 실행문으로 변환해 실행시키기 때문에 문자를 출력하는 게 아니라 문자의 실행 결과를 반환합니다.

- 이미지 출력


```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title></title>
    <script>
      function changeImage(){
        let sel = document.getElementById("sel");
        let img = document.getElementById("myImg");
        img.onload = function(){ //이미지 크기 출력용
          let mySpan = document.getElementById("mySpan");
          mySpan.innerHTML = img.width + "x" + img.height;
        }
        let index = sel.selectedIndex //내가 선택한 과일
        img.src = sel.options[index].value;
      }
    </script>
  </head>

  <body onload="changeImage()">
    <h3>이미지 출력</h3><hr>
    <form>
      <select id="sel" onchange="changeImage()">
        <option value="img\1.png">apple</option>
        <option value="img\2.png">banana</option>
        <option value="img\3.png">mango</option>
      </select>
      <span id="mySpan">이미지 크기</span>
    </form>
    <p></img></p>
  </body>
</html>
```


이미지 출력

apple ▼ 99x124


이미지 출력

banana ▼ 125x98


이미지 출력

mango ▼ 90x111


- sel.selectedIndex :
- sel.options[index].value :
-

- onfocus / onblur

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <script>
      function checkFilled(obj){
        if(obj.value == ""){
          obj.focus(); //값이 없으면 포커스로 커서를 대기
        }
      }
    </script>
  </head>
  <body onload="document.getElementById(name).focus();">
    <h3>onfocus / onblur </h3><hr>
    <p>이름 미입력시 다른 곳으로 이동 불가</p><br>
    <form>
      이름 : <input type="text" id="name" onblur="checkFilled(this)"><p></p>
      학번 : <input type="text" id="name">
    </form>
  </body>
</html>
```

onfocus / onblur

이름 미입력시 다른 곳으로 이동 불가

이름 :

학번 :

- onfocus : 자동으로 설정된 초기의 위치값
- onblur :
- focus

- 라디오 버튼

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <script>
      function findChecked(){
        let found = null; //선택한 도시를 메시지 창으로 출력하기 위해
        let kc = document.getElementsByName("city");
        for (let i=0; i<kc.length; i++){
          if(kc[i].checked == true)
            found = kc[i];
        }
        if(found != null)
          alert(found.value + "선택됨");
        else
          alert("미선택");
      }
    </script>
  </head>
  <h3>라디오 버튼</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="suwon">수원
    <input type="button" value="find checked" onclick="findChecked()">
  </form>
</body>
</html>
```

라디오 버튼

☒ 서울 ☐ 부산 ☐ 수원

127.0.0.1:5500 내용:

seoul선택됨