# 시크릿코드 샘플 예제

## 케이스 주제

버튼을 클릭하면 컨텍스트 메뉴가 나타나고, 메뉴를 선택하거나 그 외 부분을 클릭하면 사라지는 팝오버 컴포넌트를 구현하세요.

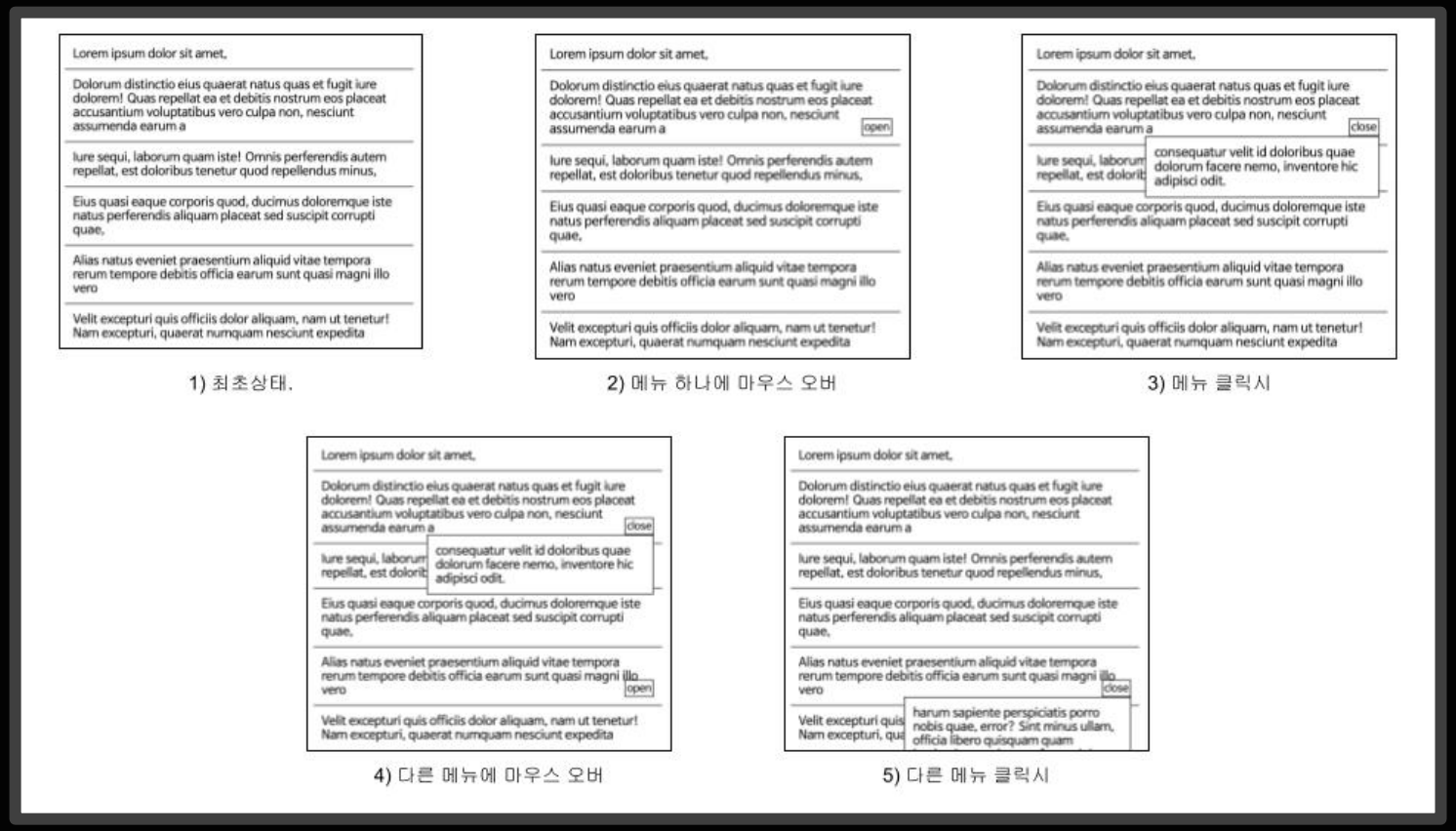
## 기능 요구사항

1. 임의의 메뉴 목록에 대하여 각 메뉴를 클릭하면 해당 메뉴에 관련된 문구가 떠있는 형태(팝오버)로 나타난다.
2. 팝오버 상태에서 해당 문구를 클릭하면 해당 메뉴가 사라진다.
3. 팝오버 상태에서 메뉴 목록을 제외한 나머지 부분을 클릭하면
4. 팝오버 상태에서 다른 메뉴를 클릭하면 해당 메뉴의 팝오버가 나타난다. 이 때 기존의 팝오버는 사라진다.

**-> 뒷장의 기능 작동 이미지를 확인하세요.**

# 시크릿코드 샘플 예제

## 기능 작동 이미지




-> 뒷장의 문제를 확인하세요.

# 시크릿코드 샘플 예제

## 문제

q1. 문제 상황에 대하여 Java Script로 동작을 구현시킬 수 있는 코드를 작성해보세요
q2. 문제 상황에 대하여 jquery로 동작을 구현시킬 수 있는 코드를 작성해보세요
q3. 문제 상황에 대하여 HTML(detail 태그)-JavaScript로 동작을 구현시킬 수 있는 코드를 작성해보세요
q4. 문제 상황에 대하여 HTML(detail 태그)-Jquery로 동작을 구현시킬 수 있는 코드를 작성해보세요
q5. 문제 상황에 대하여 React로 동작을 구현시킬 수 있는 코드를 작성해보세요
q6. 문제 상황에 대하여 React-CreatePortal 기능으로 동작을 구현시킬 수 있는 코드를 작성해보세요

## 주요 학습 키워드

• 이벤트 리스너, 이벤트 위임 구현 원리, detail 태그, React-Create Portal 등

**-> 뒷장에 시크릿코드가 이어집니다.**

# 시크릿코드 샘플 예제

## Q1. Javascript로 동작을 구현시킬 수 있는 코드를 작성해보세요

* 샘플로 해당 케이스의 한 개 문제에 대한 코드 및 간략한 해설 내용을 공유 드립니다. 한 케이스는 이러한 하위 문제 4~5개 정도로 구성되어 있습니다.

```javascript
// Write Javascript code here!
const wrapper = document.querySelector('.wrapper');
const items = document.querySelectorAll('.item');

wrapper.addEventListener('click', function (e) {
  const targetElem = e.target;
  e.stopPropagation();
  if (!targetElem.classList.contains('item')) return;

  targetElem.classList.toggle('open');
  items.forEach(cont => {
    if (cont !== targetElem) cont.classList.remove('open');
  });
});

document.body.addEventListener('click', function (e) {
  if (e.target.classList.contains('context')) return;
  items.forEach(cont => {
    cont.classList.remove('open');
  });
});
```

- 초보자들은 DOM 제어와 관련하여 이벤트 리스너를 잔뜩 작성하곤 한다. 목록의 각 아이템 하나하나마다 등록하는 식이다. 본 문제의 경우를 예로 들면 다음과 같이

items.forEach(function(item) {

item.addEventListener('click', ...)

}

이렇게 작성하면 크게 두가지 문제가 있다.

(1) 이벤트 감시대상이 많은 만큼 메모리에 부담이 된다.

(2) 어떤 변경에 의해 목록에 아이템이 추가될 경우 해당 아이템은 감시대상에 속하지 않아 팝오버 동작이 이뤄지지 않는다. 따라서 새로운 아이템이 추가될 때마다 그에 대한 리스너를 등록해주어야 한다.

**-> 뒷장에 해설이 이어집니다.**

# 시크릿코드 샘플 예제

## Q1. Javascript로 동작을 구현시킬 수 있는 코드를 작성해보세요

* 샘플로 해당 케이스의 한 개 문제에 대한 코드 및 간략한 해설 내용을 공유 드립니다. 한 케이스는 이러한 하위 문제 4~5개 정도로 구성되어 있습니다.

```javascript
// Write Javascript code here!
const wrapper = document.querySelector('.wrapper');
const items = document.querySelectorAll('.item');

wrapper.addEventListener('click', function (e) {
  const targetElem = e.target;
  e.stopPropagation();
  if (!targetElem.classList.contains('item')) return;

  targetElem.classList.toggle('open');
  items.forEach(cont => {
    if (cont !== targetElem) cont.classList.remove('open');
  });
});

document.body.addEventListener('click', function (e) {
  if (e.target.classList.contains('context')) return;
  items.forEach(cont => {
    cont.classList.remove('open');
  });
});
```

반면 리스너를 상위 노드에 등록하면 위 두가지 문제가 모두 해결된다. 따라서 리스너 등록은 최소화하는 것이 바람직하다.
이벤트 핸들러를 최소화하기 위해서는 캡쳐링/버블링을 이해하는 것이 필요하다. 나아가 리스너 함수의 첫번째 인자인 event 객체의 내부에 어떤 정보가 들어있는지를 알 필요가 있다.

- stopPropagation과 preventDefault를 구분하여 사용할 수 있는지도 중요한 요소

- (추가로 생각해볼 문제) click 이벤트의 감시 대상을 더욱 줄일 수는 없을까? 그럴 경우의 장단점은?

# 시크릿코드 이용 시 주의사항

*** 콘텐츠 이용 관련 주의사항**

- 전달받은 케이스 문제 및 해설지의 지적재산권은 패스트캠퍼스에 있습니다.
- 무단 배포 및 공유할 시 법적 문제까지 이어질 수 있습니다.
- 본 과정 수강생 분들은 학습하신 코드와 해설지에 대하여 재배포하는 일 없이 본인만 소유해주시기 바랍니다.
- 개인 블로그 및 공개된 깃헙 저장소 등 외부 채널에도 노출되지 않게 주의해주시기 바랍니다.

감사합니다