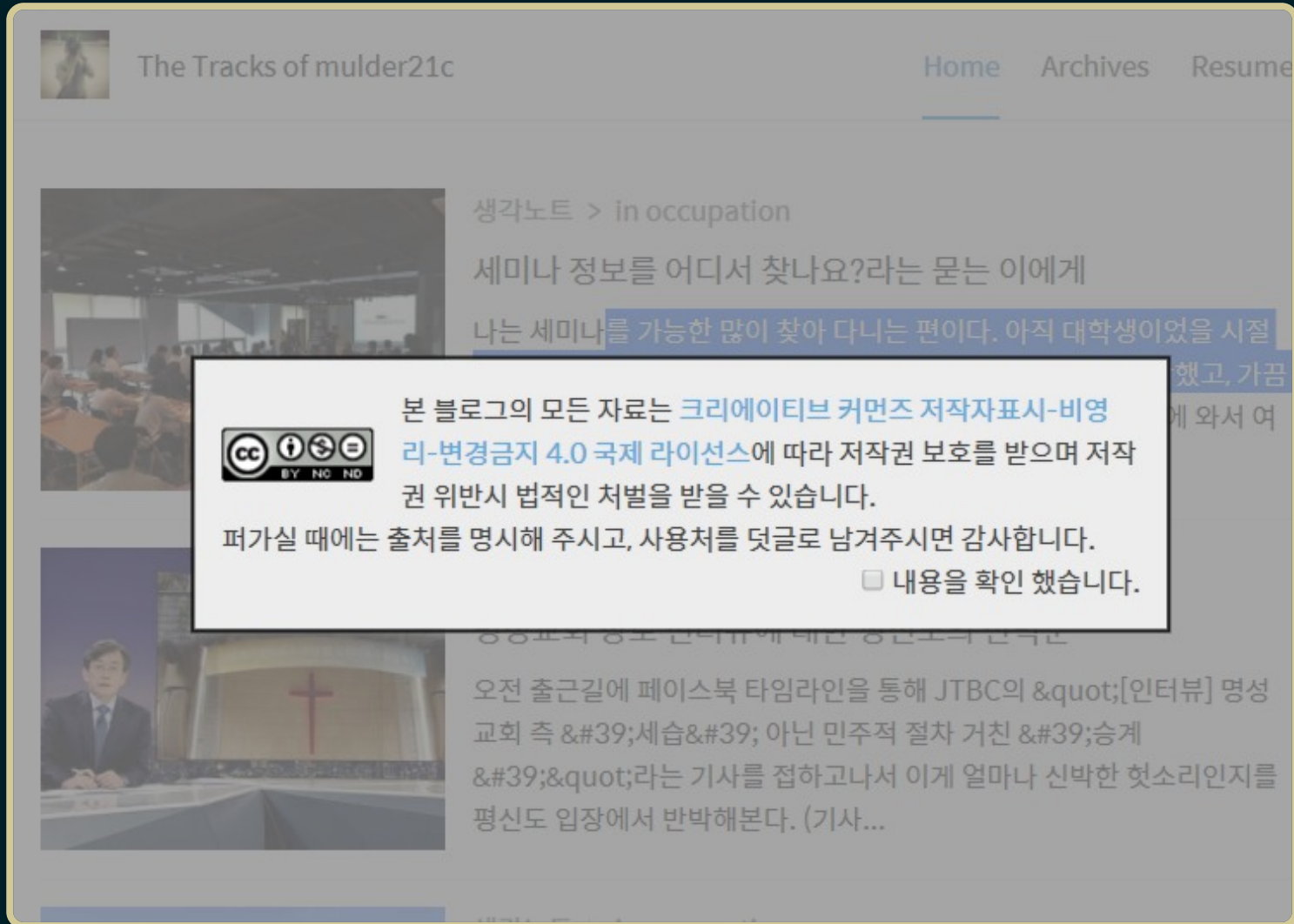


접근 가능한 레이아웃팝업

Feat. WAI-ARIA

콘텐츠연합플랫폼
클라이언트개발부 지성봉

Modal Window



The screenshot shows a web browser displaying a blog page. At the top left, there is a profile picture and the name "The Tracks of mulder21c". To the right are navigation links: "Home", "Archives", and "Resume". The main content area features a post titled "생각노트 > in occupation" with a sub-header "세미나 정보를 어디서 찾나요?라는 묻는 이에게 나는 세미나를 가능한 많이 찾아 다니는 편이다. 아직 대학생이었을 시절". A modal window is overlaid on the text, containing a Creative Commons license logo (CC BY-NC-ND 4.0) and the following text: "본 블로그의 모든 자료는 크리에이티브 커먼즈 저작자표시-비영리-변경금지 4.0 국제 라이선스에 따라 저작권 보호를 받으며 저작권 위반시 법적인 처벌을 받을 수 있습니다. 허가할 때에는 출처를 명시해 주시고, 사용처를 댓글로 남겨주시면 감사합니다. 내용을 확인 했습니다." Below the modal window, another post is partially visible with a photo of a man in a suit and a church interior, and the text "오전 출근길에 페이스북 타임라인을 통해 JTBC의 "[인터뷰] 명성교회 측 '세습' 아닌 민주적 절차 거친 '승계 '"라는 기사를 접하고나서 이게 얼마나 신박한 헛소리인지를 평신도 입장에서 반박해본다. (기사..."

Modal Window

사용자 인터페이스 디자인 개념에서 자식 윈도우에서 부모 윈도우로 돌아가기 전에 사용자의 상호동작을 요구하는 창. 응용 프로그램의 메인 창의 작업 흐름을 방해한다.

Native HTML의 한계점

- 팝업이 뺏다라는 정보를 인지할 수 없다.
- 팝업 이외의 문서 정보에 접근이 된다.
- 키보드 tab키 운용이 팝업을 벗어난다.
- 키보드 트랩 문제 (IE8 ~ 10)

Requirement

- 팝업이 열렸을 때 팝업 내용 인식 가능
- 팝업 아래의 Windows는 비활성화
- tab키 운용이 팝업 내부에서만 순환
- 팝업이 닫혔을 때 초점이 원래 있던 곳으로 반환

How To?

Step 1. 콘텐츠 역할 정의

```
role="dialog"
```

사용자가 정보를 입력하거나 응답할 것을 요구하도록 유도하기 위해
어플리케이션의 현재 처리를 중단시키도록 설계된 어플리케이션 윈도우

```
<div class="popup-wrap">
  <div class="popup-body"
    role="dialog"
    aria-labelledby="pop-title">
    <strong id="pop-title" class="modal-header">접근 가능한 레이어 팝업</strong>
    <div class="modal-body">
      <p>
        접근 가능한 레이어 팝업이란?
      </p>
      ...
    </div>
  </div>
</div>
```


Step 2. 팝업이 열릴 때 내용 인식

대화 상자 내부로 초점 이동

- 모든 상황에서 초점은 대화 상자 안에 있는 요소로 이동
- 첫 번째 초점을 얻을 수 있는 요소로 이동하는 것이 기본
- - 첫번째 포커스 가능한 요소로 초점을 이동시키는 것이 콘텐츠의 시작 부분을 스크롤 밖으로 밀어낼 경우
 - 대화상자 안에 초점을 받을 수 있는 요소가 없을 경우 정적 요소에 `tabindex="-1"` 을 추가하여 이 요소로 초점 이동

```
<div class="popup-wrap">
  <div class="popup-body" role="dialog" aria-labelledby="pop-title">
    <a class="placeholder" tabindex="-1"></a>
    <strong id="pop-title" class="modal-header">접근 가능한 레이어 팝업</strong>
    <div id="popup-contents" class="modal-body">
      <p>
        접근 가능한 레이어 팝업이란?
      </p>
    </div>
  </div>
</div>
```

```
function openPopup () {  
  var popupBody = document.querySelector(".popup-body");  
  document.documentElement.classList.add("on-popup");  
  popupBody.querySelector(".placeholder").focus();  
}
```

Step 3. 팝업 아래 Windows 비활성화

- Browse mode 진입 차단
- 대화 상자 내에서 tab 이동 순환

Browse Mode 진입 차단

`role="dialog"` 가 자동으로 처리

단, NVDA 2017.4+, NVDA 2017.2 + FireFox, JAWS 18+ 등에서만 지원

`aria-modal="true"` in ARIA 1.1

iOS 10.x/10.2에서는 문제가 있는 것으로 리포트 됨

(대화상자 제목과 지시사항들이 읽는 순서에 따라 접근 가능하지 않게 되는 문제 발생)

Browse Mode 진입 차단

차선책: 대화 상자 외 타 콘텐츠에 `aria-hidden="true"` 설정

단, 마크업 순서에 따라 적용이 어려워지는 상황이 발생.

가급적 dialog 요소를 level 1 수준으로 위치시키는 것이 정신건강에 좋음

```
function setSiblingsHidden(currElem){
  var ommits = ["script", "meta", "link", "style", "base"];
  for(var i = -1, node; node = currElem.parentNode.children[++i];){
    if(node == currElem || ommits.indexOf(node.tagName.toLowerCase()) > -1)
      continue;
    node.setAttribute("aria-hidden", "true");
    node.setAttribute("data-outside-modal", "true");
  }
}
```

```
function openPopup(){
  var popupBody = document.querySelector(".popup-body");
  document.documentElement.classList.add("on-popup");
  setSiblingsHidden(document.querySelector(".popup-wrap"));
  popupBody.querySelector(".placeholder").focus();
}
```

```
function unsetSiblingsHidden(currElem){
  for(var i = -1,
      node,
      outsides= document.querySelectorAll("[data-outside-modal]");
      node = outsides[++i]; ) {
    node.removeAttribute("aria-hidden");
    node.removeAttribute("data-outside-modal");
  }
}

function closePopup(event){
  event = event || window.event;
  document.documentElement.classList.remove("on-popup");
  unsetSiblingsHidden();
}
```


대화 상자 내에서 Tab 이동 순환

- Tab 키
 - 대화상자 내 다음 tabbable 요소로 이동
 - 마지막 tabbable 요소에 있는 경우 포커스를 대화상자 내 첫 번째 tabbable 요소로 이동
- SHFIT + Tab 키
 - 대화상자 내 이전 tabbable 요소로 이동
 - 첫번째 tabbable 요소에 있는 경우 포커스를 대화상자 내 마지막 tabbable 요소로 이동

```
(function(){
  var focuslock = (function(){
    var firstElem, lastElem;
    return {
      setFirstBtn : function(el){
        firstElem = el;
      },
      setLastBtn : function(el){
        lastElem = el;
      },
      focuslockKeyDown : function(event){
        event = event || window.event;
        var keycode = event.which || event.keyCode;
        if(event.shiftKey && keycode === 9 && event.target === firstElem){
          event.preventDefault ? event.preventDefault() : event.returnValue = false;
          lastElem.focus();
        }else if(!event.shiftKey && keycode === 9 && event.target === lastElem){
          event.preventDefault ? event.preventDefault() : event.returnValue = false;
          firstElem.focus();
        }
      }
    };
  })();
  window.focuslock = window.focuslock || focuslock;
})();
```

```
function openPopup(){
  var popupBody = document.querySelector(".popup-body");
  document.documentElement.classList.add("on-popup");

  focuslock.setFirstBtn(btnClosePopup);
  focuslock.setLastBtn(btnClosePopup);
  popupBody.addEventListener("keydown", focuslock.focuslockKeyDown);

  setSiblingsHidden(document.querySelector(".popup-wrap"));
  popupBody.querySelector(".placeholder").focus();
}
```

```
function closePopup(event){
  event = event || window.event;
  var popupBody = document.querySelector(".popup-body");
  document.documentElement.classList.remove("on-popup");
  popupBody.removeEventListener("keydown", focuslock.focuslockKeyDown);
  unsetSiblingsHidden();
}
```

Step 4. 키보드 트랩 방지

팝업이 열릴 때 초점이 얻어진 요소를 기억해 두었다가 팝업이 닫힐 때 해당 요소에 다시 초점 이동

```
(function () {  
  var focusedElem = null;  
  var btnOpenPopup = document.getElementById("open-popup");  
  var btnClosePopup = document.getElementById("close-popup");  
  ...  
  function openPopup(){  
    var popupBody = document.querySelector(".popup-body");  
    document.documentElement.classList.add("on-popup");  
    focusedElem = this;  
  
    focuslock.setFirstBtn(btnClosePopup);  
    focuslock.setLastBtn(btnClosePopup);  
    popupBody.addEventListener("keydown", focuslock.focuslockKeyDown);  
    ...  
  }  
  function closePopup(event){  
    event = event || window.event;  
    var popupBody = document.querySelector(".popup-body");  
    ...  
    focusedElem.focus();  
  }  
})();
```

Step 5. ESC 키를 눌렀을 때 팝업 닫기

```
function openPopup(){
  ...

  document.addEventListener("keydown", closePopup);
}

function closePopup(event){
  event = event || window.event;
  if(event.type === 'keydown' && event.keyCode !== 27){
    return;
  }

  ...
  document.removeEventListener("keydown", closePopup);
}
```


Used ARIA Role, Property

Roles/Property	Description
dialog	사용자가 정보를 입력하거나 응답할 것을 요구하도록 유도하기 위해 어플리케이션의 현재 처리를 중단시키도록 설계된 어플리케이션 윈도우
aria-label aria-labelledby	해당 객체의 label(이름)을 설정
aria-describe aria-describedby	해당 객체에 대한 설명 추가

Keyboard Interaction

Key

Behavior

Tab

- 대화 상자 내 다음 tabbable 요소로 초점 이동
- 마지막 tabbable 요소에 있는 경우 포커스를 대화상자 내 첫 번째 tabbable 요소로 이동

Shift + Tab

- 대화 상자 내 이전 tabbable 요소로 초점 이동
- 첫 번째 tabbable 요소에 있는 경우 포커스를 대화상자 내 마지막 tabbable 요소로 이동

Esc

대화상자 닫기

Reference

- SSB BART group- ARIA Dialog Role
https://labs.ssbartgroup.com/index.php/ARIA_Dialog_Role
- Dialog (Modal) Design Patterns - W3C
https://www.w3.org/TR/wai-aria-practices-1.1/#dialog_modal

감사합니다

NIAW AOA Github
<https://github.com/niawa/aoa>

`publisher@publisher.name`