

2022 IAT 컨퍼런스

W3C ARIA 표준

구자은

Director of IT Accessibility
University of Illinois Chicago

2022년 10월 27일

Disclaimer

제가 일리노이대학의 월드와이드웹 콘소시엄(W3C) 대표로서 W3C ARIA워킹그룹 멤버와 ARIA Authoring Practice Guide 그룹의 공동의장 및 편집자를 맡고 있으나 여기에서 발표된 자료는 제 개인의 의견이며 W3C나 일리노이대학의 공식의견을 반영하지는 않습니다.

W3C ARIA 101

들어는 봤나요?

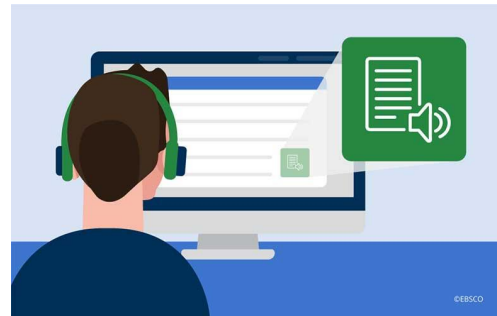
ARIA(아리아)가 무엇인가요?

- 미국 라스베가스에
유명 호텔 이름
- 오페라의 한부분
- 메이플스토리
등장인물



W3C ARIA가 무엇인가요?

- 월드와이드웹 컨소시엄(W3C) 이 정해진 절차에 따라 만든 웹접근성에 대한 표준 기술 규격
- 웹콘텐츠접근성표준(WCAG)을 넘어 다양하고 복잡한 웹어플리케이션(Rich Internet Application) 의 접근성을 도와줌
- 장애를 가진 이용자, 특히 스크린리더 이용자 및 마우스나 포인팅 기구를 이용 못하는 이용자를 돕기 위한 것.

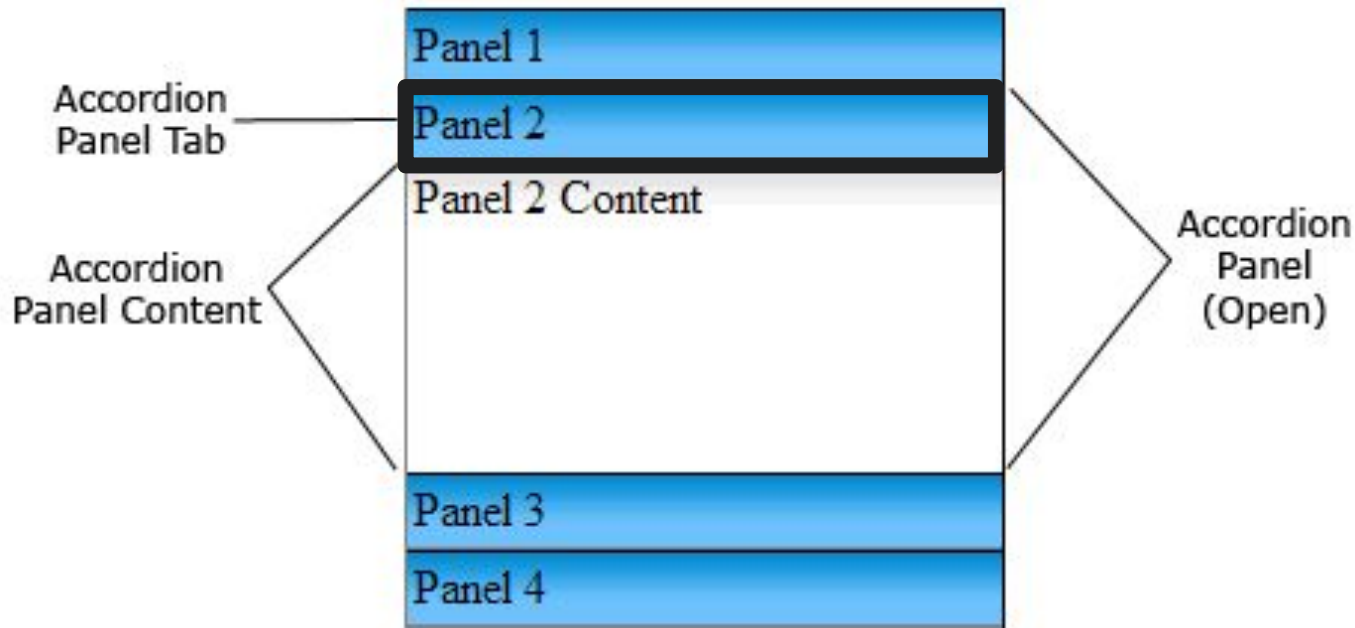


“복잡다양한 인터넷 어플리케이션” (Rich Internet Application)

이란?

- 정적인 HTML을 넘어 동적인 콘텐츠(dynamic content) 및 Ajax, HTML, JavaScript, 및 관련 기술을 이용해 개발된 사용자 인터페이스
- 다양한 위젯 패턴들- 아코디언, 다이얼로그 박스, 그리드, 리스트 박스, 슬라이드쇼 등등

어떻게 **ARIA**가 스크린 이용자를 도와주나요?(아코디언 예제)



어떻게 ARIA가 스크린 이용자를 도와주나요?

ARIA 코딩은 스크린 리더 이용자에게 “당신은 지금 보여주고 숨기기 기능을 하는 아코디언

어플리케이션 위젯이라는 것을 접근하고 있고, 이 위젯과 상호작용하기 위해서는 **엔터나 스페이스 키**를 사용할 수 있으며, 현재 선택된(**초점**이 있는) 두번째 패널 아코디언 섹션은 **열려** 있습니다.”

요약: **역할속성, 상황**의 정보를 더함으로서 스크린 리더 이용자를 도와줌

언제 HTML에서 ARIA 이용하나요?

- HTML에 피쳐는 있으나 아직 구현되지 않았거나 구현되었어도 접근성 지원([Accessibility Support](#))이 안되는 경우.
- 시각디자인 제약으로 특정 HTML을 이용할 수 없거나 HTML요소가 필요한데로 스타일 될 수 없는 경우
- 현재 피쳐가 현재 HTML에서 이용가능하지 않는 경우

ARIA의 매직파워와 위험

비상호작용적 HTML 이미지를 접근가능한 상호작용적 버튼 요소로 만드는 예시 (하지 마세요!)

Code:

```

```

*keyCode 13는 엔터키, keydown/keyup event.keyCode는 스페이스키

참조: [HTML5 Accessibility Chops: Just use a \(native HTML \) button](#)

ARIA의 매직 파워와 위험

W3C ARIA 201

뭇이 중한디?

ARIA 이용의 기본 원칙들

1. 가능하면 **Native HTML**요소나 속성이 그대로 이용.
2. ARIA 역할 속성을 더하면서 **HTML 본래속성(native semantics)**을 다른 것으로 만들지 말 것 (<h2 role="tab">heading tab</h2>)
3. 클릭, 탭, 드래그, 드랍, 슬라이드, 스크롤이 있는 위젯을 만들때는 이용자는 반드시 **키보드**를 통해 위젯을 네비게이트 하거나 같은 액션을 취할 수 있어야 함.
4. 상호작용이 가능한 모든 위젯은 접근성 이름(**accessible name**)이 있어야 함.

ARIA의 가장 중요한 원칙

- 잘못된 **ARIA** 쓰는 것 보다는 아예 **ARIA** 를 안 쓰는 것 낫다.

(NO ARIA is better than BAD ARIA!)

- 스크린 리더 이용자들에게는 눈에 보이지 않는 것을 아리아를 이용해 알려주는 것임으로 잘못된 아리아를 쓰면 오히려 더 최악의 스크린 이용자 경험을 초래 할 수 있음.

이유 1: ARIA는 숨기거나 바꾸는 역할을 할 수 있음

ARIA는 콘텐츠의 고유역할을 숨기거나 바꾸거나 할 수 있음.

- `` 스크린 이용자는 이 요소를 링크가 아니라 메뉴중 하나의 아이템으로 이해함.``
- `<a aria-label="스크린 이용자는 여기 아리아 레이블 콘텐츠만 인식할 수 있고 링크텍스트는 인식하지 못함">`링크텍스트``

이유 2: ARIA는 의미를 강화하는 역할을 할 수 있음

- 스크린 리더가 의미를 해석할 수 있도록 저자는 ARIA를 이용해 거의 모든 사용자 인터페이스를 설명할 수 있음.
- 본래 콘텐츠에 꼭 필요한 지원을 추가함.

`<button aria-pressed="false">음소거</button>`

이유 3 : ARIA 역할속성은 약속이기 때문에

ARIA는 HTML input요소와는 달리 브라우저가 키보드 액션이나 초점 스타일링을 제공하지 않으므로 한번 ARIA 역할을 정의하면 그에 따른 자바스크립트(`getElementById`, `event.keycode`)나 초점 관리(`tab index`)가 필요함.

```
<div role="button" tabindex="0" id="action">주문하기</div>
```

참고: 또 다른 자세한 예제는 이용하지 말아야 할 [ARIA를 이용해 이미지를 버튼으로 만드는 방법](#)

브라우저 및 보조기술 지원 테스트

- ARIA를 코드에 이용할 경우 보조기술의 **상호호환성 (interoperability)**를 테스트 하는 것은 아주 중요한 부분. 따라서 각각의 브라우저와 보조기술을 조합을 테스트 하는 것이 필요함 (참고 슬라이드: [ARIA Assistive Technology Testing Project](#))
- ARIA는 **절대 임시방편으로 문제를 해결하기 위한 코드방법이 아님.**

W3C ARIA 301

어떻게 쓰는지 볼까요?

접근성 이름(accessible name) 및 기술(Description)

- 저자가 정확한 접근성 이름과 적절한 기술을 제공하는 것이 접근성을 향상에 큰 도움이 됨.
- 접근성 이름(accessible name)은 보조기술이용자에게 특정 엘리먼트의 목적을 전달하고 같은 페이지에 있는 다른 엘리먼트와 구별되게 하는 역할을 함
- 접근성 기술(accessible description)은 저자가 특정 엘리먼트에 대해 추가적 정보를 제공하고자 할 때 이용함.
(예: input 필드 포맷이나 지시사항같은 정보를 제공)

접근성 이름 계산법

1. **aria-labelledby** 속성이 있으면 **aria-labelledby** 값을 이용
2. 여전히 콘텐츠 접근성 이름이 없으면 **aria-label** 값을 이용
3. 여전히 콘텐츠 접근성 이름이 없으면 **특정 호스트 언어 속성이나 엘리먼트**를 사용. HTML의 경우 엘리먼트에 따라 다름
 - a. Input type 속성이 button, submit button, reset button -> value
 - b. Input type 속성이 image button state-> img, area의 alt 속성, fieldset 인 경우 첫번째 legend 엘리먼트, 폼인 경우 label, figure인 경우 첫번째 figure caption, table인 경우 첫번째 caption 엘리먼트 등등

접근성 이름 계산의 기본 원칙

1. **ARIA** 표준에서 금지하거나 애매한 코팅패턴에 대한 주의가 필요하고, 접근성 이름이 예상한대로 여러 브라우저 결과와 매치하는지 철저하게 테스트하기
2. 보이는 글씨를 접근성 이름으로 사용 - 유지가 간편, 에러도 방지
3. 가능하면 **Native technique**을 사용(예: **HTML form**의 경우 **label element** 사용, **table** 경우 **caption**을 사용)
4. 브라우저에 생성한 접근성 이름에 의존하는 것을 피함(예: **HTML** 경우 **title and placeholder**. 이것의 목적이 접근성 이름이 아님으로 효과적이지 않은 저수준의 접근성이름을 제공하는 경향이 있음)
5. 간단하지만 유용한 이름을 만들기([효과적이고 이용하기 편한 접근성 이름 만드는 법](#))

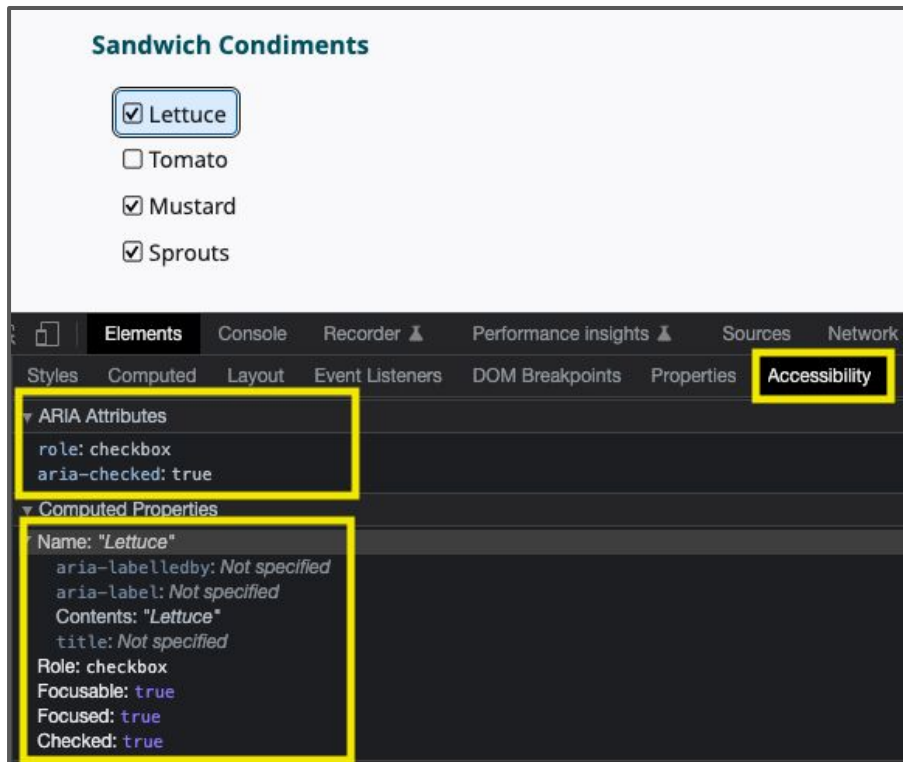
ARIA의 역할, 속성, 상황이란?

1. 역할속성(role attribute) : button, alert dialog, checkbox, dialog, grid 등 (“역할속성은 약속이다”)
2. 속성(properties): aria-labelledby, aria-checked, aria-busy, aria-control 등
3. 역동적 상태(states - dynamic properties) : aria-expanded, aria-pressed 등

참고: 가장 쉽게 기억할 수 있는 2와 3의 차이는 “역동적 상태”는 어플리케이션 개발 과정(life cycle)과정에서 변한다는 점. 그러나 이 기준이 절대적이지는 않음. 또한 2와 3도 단순히 속성(attributes)이라고 불리기도 함.

ARIA 역할속성, 역동적 상태 보는 방법들

- 구글 브라우저 접근성 익스텐션을 이용하여 ARIA역할이나 동적인 상태를 살펴 볼 수 있음.
- 샌드위치에 넣은 재료를 고를 체크박스 예제를 보면 ARIA역할이 체크박스 이고 상태가 체크 되었으며 접근성 이름은 “상추”임을 알 수 있다.
- 접근성 이름은 ARIA 레이블에서 가 아니라 콘텐츠에서 온 것이다.



ARIA 이용시 할 수 있는 실수

- 역할속성을 대문자로 쓰거나 타이포가 있음.
- 정확하지 않은 역할속성 이름.
- 정확하지 않은 ID value reference(예: aria-labelledby)
- Required owned elements가 없음(예: list 역할속성은 적어도 하나의 listitem 역할속성이 필요)
- 자식 역할속성이 없는 부모 역할 속성(또한 반대의 경우)

참고: [WAI-ARIA: Top 6 Mistakes to Avoid](#)

W3C ARIA 401

그거 들어는 봤어?

ARIA Core Specification

- [ARIA 1.2 Candidate Recommendation](#): 역할속성, 속성, 역동적 상태에 대한 존재론(ontology)를 제공하여 웹컨텐츠 및 어플리케이션의 접근성과 상호호환성을 향상시키기 위한 정보를 제공.
- [Accessible Name and Description: Computation and API Mappings](#): 어떻게 유저에이전트(user agents)가 웹에서 접근성 오브젝트(accessible objects)의 이름과 기술을 결정하는지 알려주며 이러한 정보를 접근성 API에 보여줌.
- [Core Accessibility API Mappings\(Core-aam\) 1.2](#): 어떻게 유저에이전트가 플랫폼 접근성 API에 ARIA 속성을 맵핑해야 하는지 설명

ARIA Extension - ARIA 모듈

- 전자출판(Digital Publishing) - [Digital Publishing WAI-ARIA Module](#)
- 그래픽(Graphics) - [WAI ARIA Graphics Module](#)
- Cognitive - 아직 시작 안함
- ARIA interaction module - 아직 시작 안함

Accessibility API Mapping Extension

핵심접근성(Core Accessibility) API Mapping에 근거하여 유저 에이전트가 어떻게 각각의 요소나 속성을 기술하는지 보여주는 서류들

- [전자 출판 Accessibility API Mapping](#)
- [HTML Accessibility API Mapping](#)
- [SVG Accessibility API Mapping](#)

ARIA Support Resources

- [WAI-ARIA Authoring Practice Guides](#)

ARIA의 역할속성, 상태, 역동적 상태 이용해 접근가능한 웹 콘텐츠 및 위젯을 만드는 방법을 설명하고 예제를 만들어 이해를 돕고자 함. 실제 코드 예제 및 키보드 접근법이 포함되어 있음. 보다 복잡한 예제 및 **ARIA 1.3** 예제들이 포함 될 예정.

- [Using WAI-ARIA in HTML](#)

HTML과 ARIA를 대조 비교하면서 보다 자세하게 ARIA 규칙을 설명. 상대적으로 다른 가이드라인보다 이해하기 쉬움

ARIA Assistive Technology Testing Project

- 목적: 보조기술 이용자들의 상호호환성 증진하기 위해 여러 브라우저와 스크린 리더의 조합으로 **ARIA APG** 예제를 테스트
- 한 예로 테스트 결과 보고서에 따르면 토글버튼은 **NVDA** 스크린 리더와 크롬 브라우저를 같이 이용할 때 **93%**, 죠스 스크린리더와 크롬을 이용시는 **85%**, 보이스오버와 사파리 브라우저를 이용시는 **57%**만의 **assertion**이 통과됨.

Test Plan	JAWS and Chrome	NVDA and Chrome	VoiceOver for macOS and Safari
Disclosure Navigation Menu Example	95%	None	None
Alert Example	50%	50%	50%
Command Button Example	100%	96%	88%
Color Viewer Slider	100%	71%	84%
Link Example 1 (span element with text content)	100%	96%	100%
Navigation Menu Button	98%	81%	65%
Modal Dialog Example	88%	100%	91%
Radio Group Example Using aria-activedescendant	88%	93%	83%
Toggle Button	85%	93%	57%

자료 출처 :
[ARIA AT Test Report](#)