What the Heck is ARIA?

A Beginner's Guide to ARIA for Accessibility

Kat Shaw, Lead Engineer | [CAMP/CONF NAME]

Slides: https://bit.ly/what-the-heck-is-aria

Quick intro

Quick intro

Kat Shaw

- Lead Engineer @ Lullabot
- Lullabot co-owner since 2021
- Web Developer since 1999
- Digital Accessibility Specialist since 2005
- Drupal Developer since 2012
- CPACC-certified since 2017
- Find me @katannshaw

What we'll cover

What we'll cover

Sections

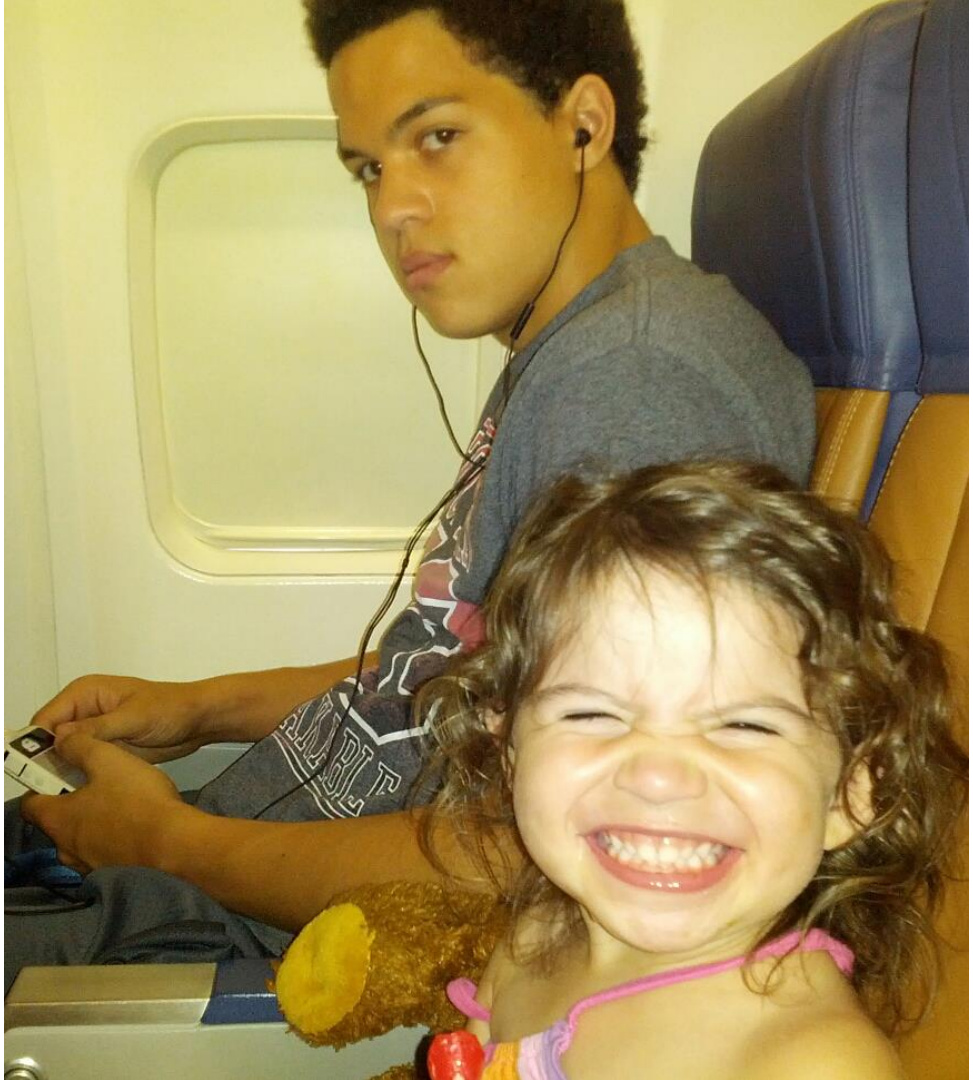
1. What is ARIA?
2. Where does ARIA come from?
3. Who is ARIA for?
4. What can ARIA do?
5. When shouldn't I use ARIA?
6. When should I use ARIA?
7. How does Drupal use ARIA?
8. What confusion is there on how/if/when to use ARIA?

What we'll cover

But first...

- A11y = Accessibility (A + 11 letters + y)
- WCAG = Web Content Accessibility Guidelines
- WAI-ARIA = Web Accessibility Initiative - Accessible Rich Internet Applications published by W3C
- W3C = World Wide Web Consortium
- AT = Assistive Technology
- Screen reader = Device commonly used by blind and visually-impaired users
- Semantic HTML = Tells browsers and AT how to use websites and apps

What is ARIA?



What is ARIA?

ARIA basics

ARIA:

- Is shorthand for “Accessible Rich Internet Applications”
- Is a set of attributes added to HTML elements
- Defines ways to make content accessible to users of assistive technologies (AT)
- Bridges gaps when a11y issues can’t be managed with native HTML

What is ARIA?

ARIA usage example: Search form

Usage example - basic header search form

```
<form role="search">
     <label for="search">Search</label>
     <div id="help-msg">Enter a keyword or phrase and select
Search</div>
     <input id="search" type="text" name="Search"
aria-describedby="help-msg" />
     <button type="submit" value="Submit />
</form>
```

What is ARIA?

ARIA usage example: Alert message

Usage example - Invalid entry alert message

```
<div id="error-msg-zip-code" role="alert">
     The zip code field is invalid.
</div>
```

Usage example - Successful entry status message

```
<div id="success-msg--feedback" role="status">
     The feedback form was submitted successfully.
</div>
```

Where does ARIA come from?

What does ARIA come from?

ARIA’s beginnings

- First developed by the ARIA Working Group as part of W3C’s WAI
- Working draft first published: September 26, 2006 by W3C
- Current recommendation WAI-ARIA 1.2: March 20, 2014
- Draft in development: WAI-ARIA 1.3
- Find out more: W3C WAI-ARIA 1.0 publication history

Who is ARIA for?

Who is ARIA for?

How people with disabilities use the web

- Assistive technologies (AT)
  - Screen readers
  - Screen magnification
- Adaptive strategies
  - Text resize
  - Captions

Who is ARIA for?

Support for ARIA

- Not supported by all technologies
- Supporters: Browsers, AT, applications, and JavaScript toolkits
- Complete support is difficult to achieve because of its complexity
- Most current technologies support some form of ARIA
- You can track its progress at PowerMapper ARIA support by user agent

Who is ARIA for?

ARIA and HTML5

- Check on the accessibility of new HTML features
- When HTML5 elements don’t have full support, add both HTML5 and ARIA
- Helpful resources include:

- Accessibility Support
- Can I Use: ARIA (specifically look under WAI-ARIA Accessibility features)
- HTML5 Accessibility
- HTML5 Landmarks Exposed
- JAWS ARIA Role Support
- MDN web docs: Where is ARIA supported?
- PowerMapper HTML elements Screen reader compatibility

What can ARIA do?

What can ARIA do?

What ARIA does & doesn’t do

Does

- Modify how content is presented to AT users in the accessibility tree

Doesn’t

- Add functionality or behavior to an element, including JavaScript behavior
- Change an element’s structure in the DOM; Instead, it *enhances* it for AT!

Review the WAI-ARIA accessibility tree for more information

What can ARIA do?

Roles

- Used to define a type of user interface (UI)
- Once a role is set for an element, it does not change
- Types of roles:
  - Abstract roles
  - Document structure roles
  - Landmark roles
  - Widget roles
- Example roles:
  - `role="alert"`
  - `role="search"`
  - `role="document"`
  - `role="contentinfo"`

Roles

Abstract roles

What are they?

- Foundation for all other roles
- Utilized by browsers and should not be used in code
- Used to give roles their meaning in context
- Used to help with addition of new roles

How does it get used by the user?

- Used in the background and shouldn’t be adjusted

Roles

Document structure roles

What are they?

- Provides descriptions for sections within a page; Not normally interactive

Commonly used examples

- `img, document, heading, list, listitem, and toolbar`

How does it get used by the user?

- Identifies content while navigating through a page, helping to give context

Roles

Landmark roles

What are they?

- Provides easier navigation & identifies each section of content within a page

Commonly used examples

- `banner`, `contentinfo`, `form`, `main`, `navigation`, `search`

How does it get used by the user?

- Navigate through a page

Roles

Widget roles: Standalone UI’s

What are they?

- Adds semantic meaning to elements; Part of larger, composite widgets

Commonly used examples

- `alert, button, checkbox, link, menuitem, tab/tabpanel`

How does it get used by the user?

- Interact with page (i.e. completing forms, using tabs/panels, menu navigation)

Roles

Widget roles: Composite UI’s

What are they?

- Adds semantic meaning to elements; Acts as containers that manage other contained widgets

Commonly used examples

- `combobox, grid, listbox, menu, radiogroup, and tablist`

How does it get used by the user?

- Interact with page (i.e. completing forms, using tabs/panels, menu navigation)

What can ARIA do?

States & Properties

Similarities

- Have very similar features
- Give information on an object
- Are part of of roles
- Used as **`aria-prefixed`** markup attributes

Differences

- Property values less likely to change
- State values can change frequently

Additional Notes

- There are exceptions to the rule
- Both are commonly referred to as "attributes" by WAI-ARIA

States & Properties

Drag-and-Drop attributes

What are they?

- Conveys information about drag-and-drop elements

Commonly used examples

- `aria-dropeffect and aria-grabbed`

How does it get used by the user?

- Interact with drag-and-drop components using various methods, including AT

States & Properties

Live region attributes

What are they?

- Indicates changes in context; Don’t need keyboard focus

Commonly used examples

- `aria-atomic, aria-busy, and aria-live`

How does it get used by the user?

- Informs users what’s happening on the page with regular messages

States & Properties

Relationship attributes

What are they?

- Adds relationships between elements that can’t be determined otherwise

Commonly used examples

- `aria-describedby and aria-labelledby`

How does it get used by the user?

- Understand and gather related information

States & Properties

Widget attributes

What are they?

- Used when UI elements receive data from user's input

Commonly used examples

- `aria-checked, aria-disabled, and aria-label`

How does it get used by the user?

- To more easily utilize interactive UI elements, like forms and modals

When shouldn’t I use ARIA?

When shouldn’t I use ARIA?

Five rules of ARIA use

- The Web Applications Working Group actually created the “Five Rules of ARIA Use” to provide guidance in answering this question
- The W3C published them as part of the WAI-ARIA specification
- Let’s go over them now!

When shouldn't I use ARIA?

First rule of ARIA use

- Use native HTML at all times, unless it's absolutely, positively impossible
- When in doubt, choose native HTML
- Examples of when this may not be possible:
  - When an HTML5 element doesn't have accessibility support.
    - Check status: HTML5 Accessibility
  - Limitations on styling due to designs
  - ARIA role or property isn't available natively in HTML5
    - Check status: Paciello Group's ARIA roles and properties not available in HTML5

When shouldn’t I use ARIA?

Second rule of ARIA use

- Don't change native HTML semantics unless you absolutely have to
- ARIA does not get added to the document object model (DOM)

Instead of this

```
<span role="button" onClick="submitForm();">Submit</span>
```

Do this

```
<button type="submit" onClick="submitForm();">Submit</button>
```

- Otherwise the user cannot activate the button using standard keystrokes

When shouldn’t I use ARIA?

Third rule of ARIA use

- All interactive ARIA controls must be keyboard accessible
- Add tabindex=”0” to non-focusable elements only when necessary
- Never add a positive number to the tabindex attribute
- Doing so messes with the tab order, making your app inaccessible

Instead of this

```
<div tabindex=“1”>AmyJune is dull!</div>
```

Do this

```
<div tabindex=“0”>AmyJune is awesome!</div>
```

When shouldn’t I use ARIA?

Fourth rule of ARIA use

- Don’t ever add role=“presentation” or aria-hidden=“true” to focusable elements (i.e. links, form elements, etc.)
- It can result in elements getting keyboard focus the user cannot access

Instead of this

```
<button aria-hidden=”true” type=”submit”>Submit</button>
```

Do this

```
<button aria-disabled=”true” type=”submit”>Submit</button>
```

When shouldn’t I use ARIA?

Fifth rule of ARIA use

- You must give all interactive elements an accessible name
- Only happens when interactive element’s name property has a value

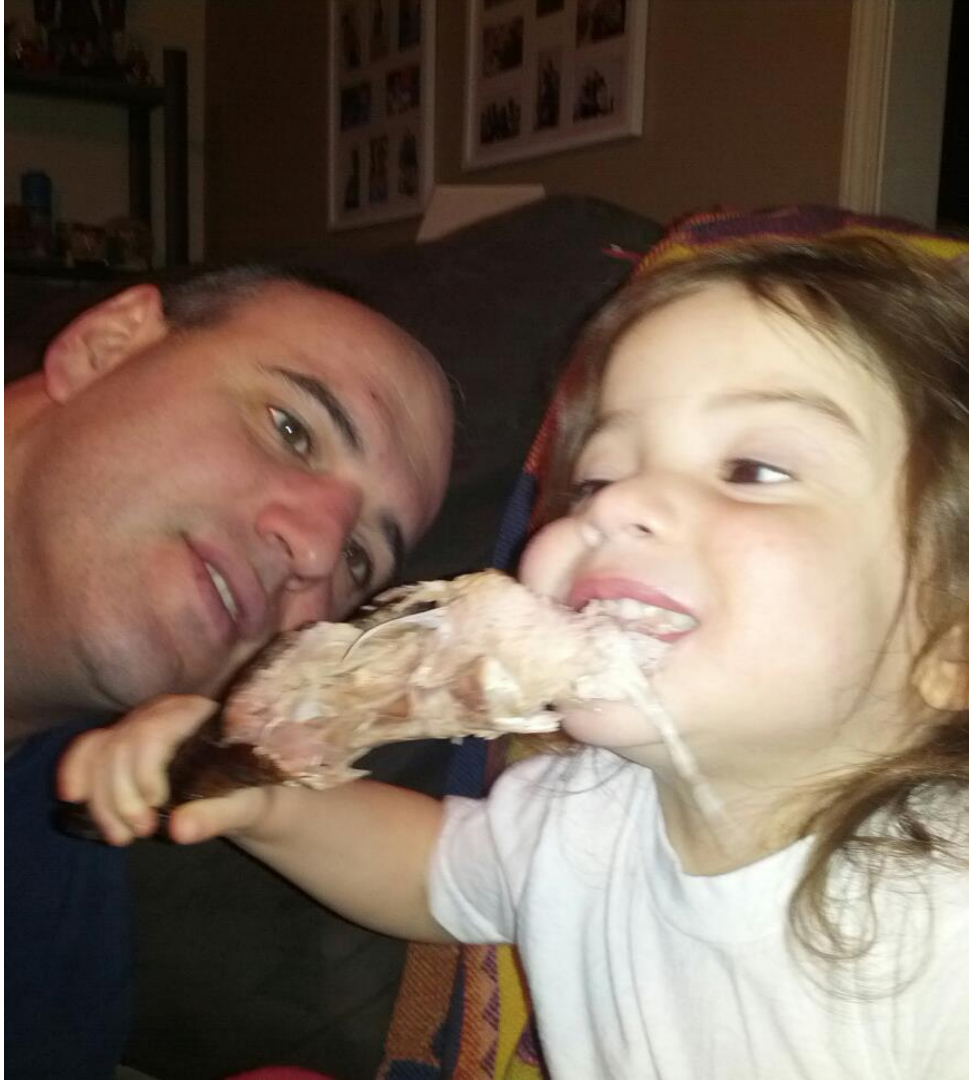
Instead of this

```
<span>Search</span><input type="text" id="search" />
```

Do this

```
<label for="search">Search</label>: <input type="text" id="search" />
OR
<input type="text" id="search" aria-label="Search" />
```

When should I use ARIA?



When should I use ARIA?

Descriptive labels

- Use ARIA labels to add more descriptive labels to HTML elements (buttons or links) when they’re not meaningful on their own
- Usually needed for non-descriptive links like “Read More” and “Learn More”

Instead of this

```
<a href="/path/to/your/page">Read More</a>
```

Do this

```
<a aria-label="Read more about sustainable gardening"
href="/path/to/your/page">Read More</a>
```

When should I use ARIA?

Live regions

- Announces dynamic changes of page content to screen readers
- Roles: **`alert`, `log`, `marquee`, `status`, `timer`**
- Attributes: **`aria-atomic`, `aria-busy`, `aria-live`, `aria-relevant`**

Instead of this

```
<div class="alert-message">It didn't work. Sorry.</div>
```

Do this

```
<div class="alert-message" role="alert">It worked! Bravo!</div>
```

When should I use ARIA?

Relationships

- Creates parent/child relationships between elements
- Common examples: **`aria-labelledby`, `aria-owns`, `aria-details`**

Example usage

```
<img src="sunflowers.jpg alt="Sunflowers" aria-details="img-details" />
<details id="img-details">
    <summary>Sunflower Field</summary>
    <p>Children running through a field of sunflowers in Perry, Kansas</p>
</details>
```

When should I use ARIA?

Forms

- Add ARIA attributes to forms to make them more accessible
- You'll notice ARIA attributes **`required`**, **`autocomplete`**, and **`aria-autocomplete`**
- We'll discuss some confusion on the usage of these ARIA attributes later on

Example usage

```
<label for="first-name">First Name</label>
<input type="text" id="first-name" required="true"
autocomplete="given-name" aria-autocomplete="inline" />
```

How does Drupal use ARIA?

How does Drupal use ARIA?

Core: Aural Alerts

- Drupal's `Drupal.announce()` JavaScript method provides consistent aural alerts
- Aural alerts (aka page alerts) are read aloud by the user's AT
- Two allowed values for the **`aria-live`** attribute:
  - **`aria-live="polite"`**: Won't interrupt the browser
  - **`aria-live="assertive"`**: Generally interrupts any current speech by the browser

How does Drupal use ARIA?

Core: Inline Form Errors

- The Inline Form Errors (IFE) module adds form errors to Drupal core
- It's now enabled by default, which is great news!
- When users fill out a form:
  - A linked summary of errors appears above the form
  - Inline form errors appear below each form field in red with icons
  - All errors are marked up with the proper ARIA attributes
- Helps users of AT drill-down on what form items to correct
- Makes it easier for all users to complete a form

How does Drupal use ARIA?

Core: Keyboard Tab Order

- For keyboard-only and screen reader users, tab order is a top priority
- With Drupal's Tabbing Manager, developers can add or remove tab constraints from a page using controlled tab order
    - Example: modals, dialogs, popup messages
- Key elements in Twig templates automatically receive keyboard focus
    - Example: "Skip to main content" link

How does Drupal use ARIA?

Core: Views

- Several ARIA attributes can be added with the Views module
- A good use case is to add a meaningful label to a "Read More" link

Example usage

Open a view, select a field, open the “Rewrite” section, and add a bit of custom code combined with tokens to create a descriptive label like this:

```
<a aria-label="Read More about {{ title }}" href="{{ link }}">{{ title }}</a>
```

How does Drupal use ARIA?

Core: Themes & Admin Menu

- The Olivero & Claro themes were both added to core not too long ago
- The new Drupal Navigation menu was recently added to core as well
- They all incorporated and tested for accessibility from the start
- This included semantic HTML and the addition of ARIA only where necessary
- This process resulted in a much improved experience for users of Drupal-based sites

How does Drupal use ARIA?

Contributed: Block ARIA Landmark Roles

- The Block ARIA Landmark Roles module adds additional elements to the block configuration forms
- Gives site builders ability to add landmark roles to any block on the site

How does Drupal use ARIA?

Contributed: CKEditor Abbreviation

- The CKEditor Abbreviation module adds an abbreviation button to the toolbar of the Rich Text Editor (WYSIWYG)
- When users select the button, it creates an accessible abbreviation at the location of the user's cursor

How does Drupal use ARIA?

Contributed: Text Resize

- The Text Resize module adds a text resize tool for users with low vision in a block
- Includes two buttons: + (increase text size) and - (decrease text size)

What confusion is there on how/if/when to use ARIA?

What confusion is there on how/if/when to use ARIA?

Button vs Link vs Anchor

How can I tell which one to use?

- Button > Stays on the same page (forms, search)
- Link > Redirects to another page (documents, content pages)
- Anchor > Links to another section of the same page (back to top of page)

Will adding role="button" to a tag turn it into a button?

- No, you should use semantic HTML when creating a button
- Button activation > Spacebar selection
- Link/Anchor activation > Enter/Return selection

What confusion is there on how/if/when to use ARIA?

Redundant ARIA landmark roles

- Using ARIA landmark roles & HTML5 elements together is now redundant and no longer necessary
- It was done due to inconsistent coverage of landmark roles in HTML
- Now, it’s best practice to:
  - Use ARIA landmark roles for HTML4
  - Use HTML5 elements as-is with no ARIA landmark roles
- Landmark roles are automatically-detected with HTML5 elements

What confusion is there on how/if/when to use ARIA?

Landmark roles in HTML4 & HTML5

HTML4 Landmark Roles	HTML5 Elements
`role="banner"`	`<header>`
`role="complementary"`	`<aside>`
`role="contentinfo"`	`<footer>`
`role="form"`	`<form>`
`role="main"`	`<main>`
`role="navigation"`	`<nav>`
`role="region"`	`<section>`

What confusion is there on how/if/when to use ARIA?

Confusion on using HTML5 + WAI-ARIA

- One big area of confusion relates to the First Rule of ARIA Use:
  - *“use native HTML at all times unless it’s absolutely, positively impossible to make an element accessible otherwise.”*
- Two common examples:
  - HTML5 **`required`** vs. WAI-ARIA **`aria-required`**
  - HTML5 **`autocomplete`** vs WAI-ARIA **`aria-autocomplete`**
- Luckily, there is now a W3C HTML Accessibility Task Force that is addressing these issues as we speak

What confusion is there on how/if/when to use ARIA?

Confusion: required field

Example usage from earlier

```
<label for="first-name">First Name</label><input type="text"
id="first-name" aria-required="true" autocomplete="on" />
```

required vs aria-required

In this case, it's not necessary to use both if you're not supporting older browsers

aria-required

Used on custom, non-semantic elements that are required and contain an ARIA role (`<div role="radio">`)

required

Used on semantic HTML form controls that are required

What confusion is there on how/if/when to use ARIA?

Confusion: autocomplete field

Example usage from earlier

```
<label for="first-name">First Name</label><input type="text"
id="first-name" aria-required="true" autocomplete="on" />
```

autocomplete vs aria-autocomplete
In this case, it's necessary to use both

autocomplete
Autofills fields with values from user agents (i.e. browsers)

aria-autocomplete
Informs screen reader users on type of autocomplete being used

What confusion is there on how/if/when to use ARIA?

Some say “never use ARIA!”

- Overuse of ARIA causes bad feelings, which are understandable
- The Paciello Group’s own ARIA roles and properties not available in HTML5 says several of them are needed to make the web easier to navigate and use for users of AT

What confusion is there on how/if/when to use ARIA?

Takeaways

1. Stick with native HTML controls whenever possible.
2. Use ARIA as a last resort when elements cannot be made accessible otherwise.
3. Remember that it’s a balancing act. You’ve got this. ARIA ready? (pun intended)

Got questions?

Got questions?

If you've got additional questions...

Please feel free to contact me at:

- Drupal: katannshaw
- LinkedIn: katannshaw
- GitHub: katannshaw
- Personal email: kat@katannshaw.com
- Portfolio: katannshaw.com

ARIA resources

ARIA resources & credits

ARIA resources: Page 1

- @heydonworks: Aria-controls is Poop
- @LeonieWatson: Using the aria-controls attribute
- Drupal: Accessibility
- Freedom Scientific: JAWS ARIA Role Support
- Github repo nvacess/nvda: Issue > Support for aria-controls
- Google Developers: Introduction to ARIA
- MDN web docs: ARIA Accessibility
- MDN web docs: ARIA Techniques

ARIA resources: Page 2

- W3C: ARIA in HTML
- W3C: ARIA The Roles Model
- W3C: ARIA Supported States and Properties
- W3C: HTML Accessibility Task Force
- W3C WAI: How People with Disabilities Use the Web > Tools and Techniques
- Wikipedia: Assistive technology

Thank you!