



how to a11y

All you need to know about building accessible web content

Tanja Ulianova

Tanja Ulianova



Web Developer from Karlsruhe

all things web
a11y nerd
clean architecture



@techtanja

What is a11y?

a11y = accessibility

making web content usable for any people
(regardless of their abilities)

part of universal design and good usability

Users!



low vision, blindness, color blindness
Screen Reader

people who can't see a mouse cursor
rely on screen reader



low hearing, deafness

somebody who cannot hear well needs
captions



motor impairments
people who can't operate a mouse

keyboard access



cognitive impairments, dyslexia,
migraines

simple language, uncomplicated GUIs,
reader mode



seizure disorders

avoid flashing animations

"us" vs. "them"

not needing a11y is only temporary

natural • illness • situational • aging

Guidelines

Abstract**Status of This Document****Introduction**

- .1 Background on WCAG 2
- .2 WCAG 2 Layers of Guidance
- .3 WCAG 2.2 Supporting Documents
- .4 Requirements for WCAG 2.2
- .5 Comparison with WCAG 2.1
 - .5.1 New Features in WCAG 2.2
 - .5.2 Numbering in WCAG 2.2
 - .5.3 Conformance to WCAG 2.2
- .6 Later Versions of Accessibility Guidelines

1. Perceivable

- 1.1 Text Alternatives
 - 1.1.1 Non-text Content
- 1.2 Time-based Media
 - 1.2.1 Audio-only and Video-only (Prerecorded)
 - 1.2.2 Captions (Prerecorded)
 - 1.2.3 Audio Description or Media Alternative (Prerecorded)
 - 1.2.4 Captions (Live)
 - 1.2.5 Audio Description (Prerecorded)
 - 1.2.6 Sign Language (Prerecorded)
 - 1.2.7 Extended Audio Description (Prerecorded)
 - 1.2.8 Media Alternative (Prerecorded)
 - 1.2.9 Audio-only (Live)
- 1.3 Adaptable
 - 1.3.1 Info and Relationships
 - 1.3.2 Meaningful Sequence
 - 1.3.3 Sensory Characteristics
 - 1.3.4 Orientation
 - 1.3.5 Identify Input Purpose

Web Content Accessibility Guidelines (WCAG) 2.2



W3C Working Draft 21 May 2021

This version:

<https://www.w3.org/TR/2021/WD-WCAG22-20210521/>

Latest published version:

<https://www.w3.org/TR/WCAG22/>

Latest editor's draft:

<https://w3c.github.io/wcag/guidelines/22/>

Previous version:

<https://www.w3.org/TR/2021/WD-WCAG22-20210513/>

Latest Recommendation:

<https://www.w3.org/TR/WCAG/>

Editors:

Chuck Adams (Oracle)
Alastair Campbell (Nomensa)
Rachael Montgomery (Invited Expert)
Michael Cooper (W3C)
Andrew Kirkpatrick (Adobe)

Participate:

[GitHub w3c/wcag](#)
[File a bug](#)
[Commit history](#)
[Pull requests](#)

Copyright © 2020-2021 W3C® (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and document use rules apply.

Abstract

Web Content Accessibility Guidelines (WCAG) 2.2 covers a wide range of recommendations for making Web content more accessible. Following these guidelines will make content more accessible to a wider range of people with disabilities, including accommodations for blindness and low vision, deafness and hearing loss,

Web Content Accessibility Guidelines: <https://www.w3.org/TR/WCAG22/>

WCAG 2.2

13 Guidelines based on four principles:

perceivable, operable, understandable, robust

Success Criteria (aka Levels of Conformance)

A basic a11y features

AA worldwide legal standard ([list of laws](#))

AAA extended a11y features

<https://www.w3.org/TR/WCAG22/>

How to Meet WCAG (Quick Reference)

A customizable quick reference to Web Content Accessibility Guidelines (WCAG) 2 requirements (success criteria) and techniques.

[Show About & How to Use](#)

Contents

Filter

Hide

Selected Filters: **WCAG 2.1:** all success criteria and all techniques.

[Clear filters](#)

[+ Expand all sections](#)

[Share](#)

1. Perceivable

1.1 Text Alternatives

1.1.1 Non-text Content

1.2 Time-based Media

1.2.1 Audio-only and Video-only (Prerecorded)

1.2.2 Captions (Prerecorded)

1.2.3 Audio Description or Media Alternative (Prerecorded)

1.2.4 Captions (Live)

1.2.5 Audio Description (Prerecorded)

1.2.6 Sign Language (Prerecorded)

1.2.7 Extended Audio Description (Prerecorded)

1.2.8 Media Alternative (Prerecorded)

1.2.9 Audio-only (Live)

1.3 Adaptable

1.3.1 Info and Relationships

1.3.2 Meaningful Sequence

1.3.3 Sensory Characteristics

1.3.4 Orientation

1.3.5 Identify Input Purpose

1.3.6 Identify Purpose

Principle 1 – Perceivable

Information and user interface components must be presentable to users in ways they can perceive.

Guideline 1.1 – Text Alternatives

Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

1.1.1 Non-text Content — Level A

All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below. [Show full description](#)

[Understanding 1.1.1](#)

[Show techniques and failures for 1.1.1](#)

[SHARE](#) | [BACK TO TOP](#)

Guideline 1.2 – Time-based Media

Provide alternatives for time-based media.

1.2.1 Audio-only and Video-only (Prerecorded) — Level A

For prerecorded audio-only and prerecorded video-only media, the following are true, except when the audio or video is a media alternative for text and is clearly labeled as such: [Show full description](#)

[Understanding 1.2.1](#)

How to Meet WCAG: <https://www.w3.org/WAI/WCAG21/quickref/>

Testing for a11y

Start with a checklist

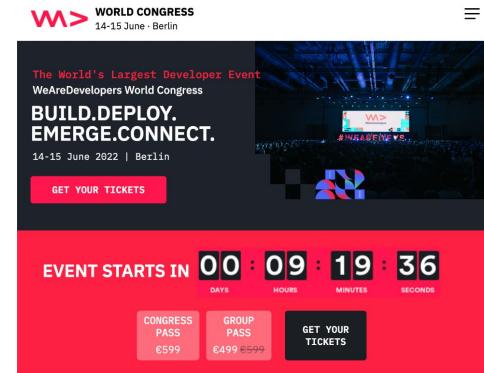
- Define your most important use cases
- Monitor from the beginning
- Or begin optimizing from most important use case

	Use Case 1	Use Case 2	Use Case 3
Acceptance Criteria 1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Acceptance Criteria 2	<input checked="" type="checkbox"/>		

A11y Project checklist: <https://www.a11yproject.com/checklist/>

WebAim checklist: <https://webaim.org/standards/wcag/checklist>

Accessibility Audits Dev Tools: Chrome



93

Accessibility

These checks highlight opportunities to [improve the accessibility of your web app](#). Only a subset of accessibility issues can be automatically detected so manual testing is also encouraged.

CONTRAST

- Background and foreground colors do not have a sufficient contrast ratio.

These are opportunities to improve the legibility of your content.

INTERNATIONALIZATION AND LOCALIZATION

- `<thead>` element does not have a `lang` attribute

These are opportunities to improve the interpretation of your content by users in different locales.

NAMES AND LABELS

- Links do not have a discernible name

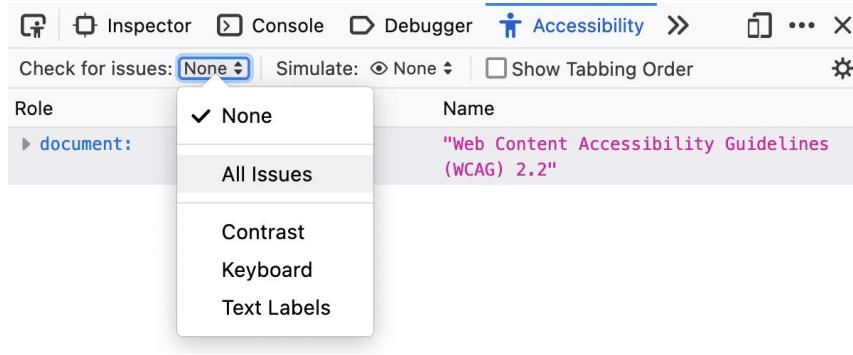
These are opportunities to improve the semantics of the controls in your application. This may enhance the experience for users of assistive technology, like a screen reader.

NAVIGATION

- Heading elements are not in a sequentially-descending order

Chrome Dev Tools > Lighthouse > Accessibility

Accessibility Audits Dev Tools



Firefox Dev Tools > Accessibility > Check for Issues



Web page address:



WAVE Web Accessibility Evaluation Tool

WAVE® is a suite of evaluation tools that helps authors make their web content more accessible to individuals with disabilities. WAVE can identify many accessibility and Web Content Accessibility Guideline (WCAG) errors, but also facilitates human evaluation of web content. Our philosophy is to focus on issues that we know impact end users, facilitate human evaluation, and to educate about web accessibility.

You can use the online WAVE tool by entering a web page address (URL) in the field above. [WAVE Chrome](#), [Firefox](#), and [Edge browser extensions](#) are available for testing accessibility directly within your web browser - handy for checking password protected, locally stored, or highly dynamic pages. We also have a [Accessibility IMpact \(AIM\) service](#), [subscription WAVE API](#), and a [stand-alone WAVE API and testing engine](#) for easily collecting data on many pages. If you need enterprise-level reporting and tracking of accessibility, WAVE powers the [Pope Tech accessibility tool](#).



WAVE Audits: <https://wave.webaim.org/>

Automated Testing

aXe Core

- CLI
- pure JavaScript API
- Wrapper (Selenium, etc..)
- Used in Lighthouse

BUT: not everything can be automated

manual tests + user studies are a necessity

pa11y

- CLI
- pure JavaScript API
- Some wrappers
- Dashboard App

pa11y dashboard - your automated accessibility testing pal



Manual Screen Reader Testing

```
document.body.style.clip = 'rect(0,0,1px,1px)';  
document.body.style.overflow = 'hidden';  
document.body.style.position = 'absolute';
```

Free Screen Readers:

Windows: NVDA (www.nvaccess.org), Narrator - native

Mac: VoiceOver - CMD + F5 (System Preferences > Accessibility > Voice Over)

Linux: Orca (<https://help.gnome.org/users/orca/stable/>)

BRLTTY (<https://brltty.app/>)

Voxin (<https://oralux.org/>)

<DEMO TIME>

Color

Guideline 1.4 – Distinguishable:

Make it easier for users to see and hear content including separating foreground from background.

Technical implementation

- Contrast!
- Don't solely rely on color for meaning or information

Contrast

Minimal requirement (AA)

4.5 : 1
for small text

3 : 1
large text (>18p, >14p bold)

Enhanced requirement (AAA)

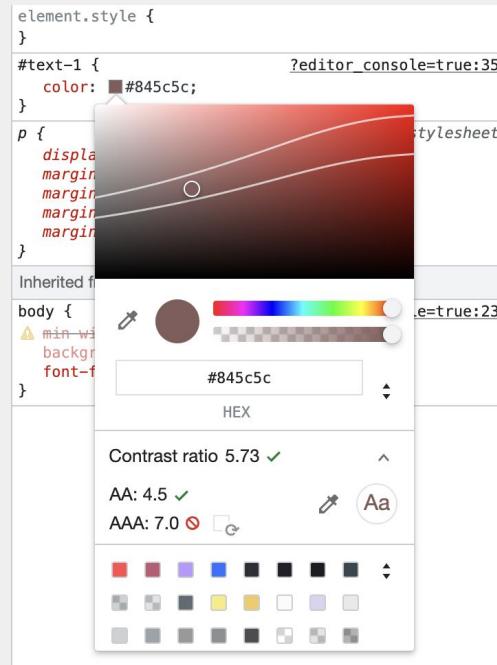
7 : 1
for small text

4.5 : 1
large text (>18p, >14p bold)

Color Contrast Checkers

Lore ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

- Chromium Dev Tools
- Online Tools:
 - [Web Aim Contrast Checker](#)
 - <https://contrast-ratio.com/>
 - ...
- Plugins for most design tools



Don't communicate through color only

Always use textual information as well

Email:

blabla

Bad

Email:

blabla

Must be a valid email

Good

<DEMO TIME>

Color a11y tools in firefox

Using HTML:

Labelling content

Guideline 1.1 – Text Alternatives:

Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

Technical implementation

- Provide labels for input fields
- alt attribute for images



key icon alt="

A couple of sheep on a
green hill looking straight
into the camera"

And if your image is just decorative...

```
<img alt="">
```

If your image stands for a control..

Focus on the action!



"Shopping Cart"

"Add to shopping cart"



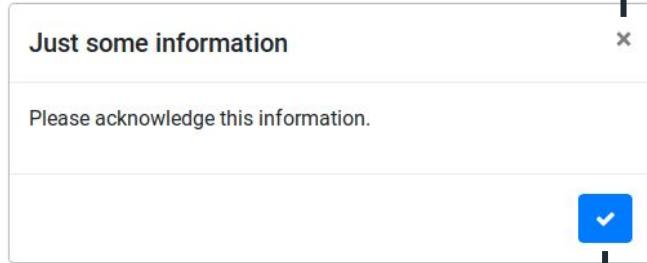
"Star"

"Add to favorites"

(Not good)

(Good)

Also label icon buttons



```
<button>x</button>
```

Bad

```
<button aria-label="Close">x</button>
```

```
<button>  
  <i class="icon-check"></i>  
</button>
```

Bad

```
<button aria-label="Confirm">  
  <i class="icon-check"></i>  
</button>
```

Form Labels

Address

1234 Main St

Address 2

Apartment, studio, or floor

City

State

Zip

Check me out

Sign in

Label every input correctly

[*wrong*]

```
<div>
  <div class="label">Name of kitten</div>
  <input type="text">
</div>
```

make it clear which label
belongs to which input field

[*correct*]

```
<div>
  <label class="label" for="kittenName">Name of kitten</label>
  <input type="text" id="kittenName">
</div>
```

Also correct ...

wrap input in label

```
<label>Name <input type="text"></label>
```

labelling only for
assistive technology

```
<input type="text" aria-label="Name" placeholder="Name">
```

multiple inputs with
one label

```
<label id="links" for="firstLink">Links</label>
<input type="text" aria-labelledby="links" id="firstLink">
<input type="text" aria-labelledby="links">
```

Using HTML:

Keyboard Support

Guideline 2.1 – Keyboard Accessible:

Make all functionality available from a keyboard.

Technical implementation

- Proper focus handling via keyboard
- Avoid keyboard traps

Sign me up

```
<div class="button" onclick=". . .>
  Sign me up
</div>
```

Antipattern!

- No focus via tab
 - tabindex="0"
- No keyboard events
 - *Trigger onclick on Enter keydown event*
- Not announced as a button
 - role="button"

Sign me up

```
<button class="button">  
  Sign me up  
</button>
```

Good! 🎉

- Extensive keyboard support for free:
 - Events
 - Focus
- Style as you wish

<button> for actions
<a> for navigation & links

Tabindex

tabindex="0"

Make an otherwise unfocusable element focusable.

Use its position in DOM for focus order.

Avoid tabindex > 0

Creates a page global focus order (all >0 in their order, before all with 0).

Very hard to manage and bad for non vision impaired keyboard users.

tabindex="-1"

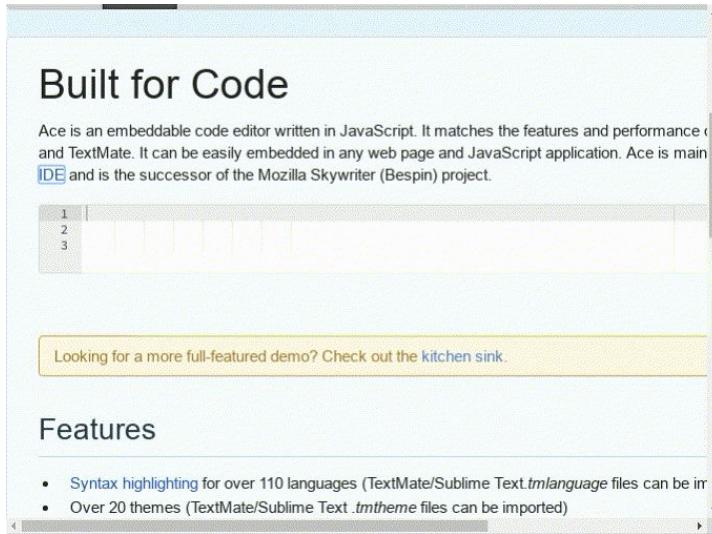
Exclude an otherwise focusable element from focus order.

Make an unfocusable element, focusable via `element.focus()`



Don't trap focus

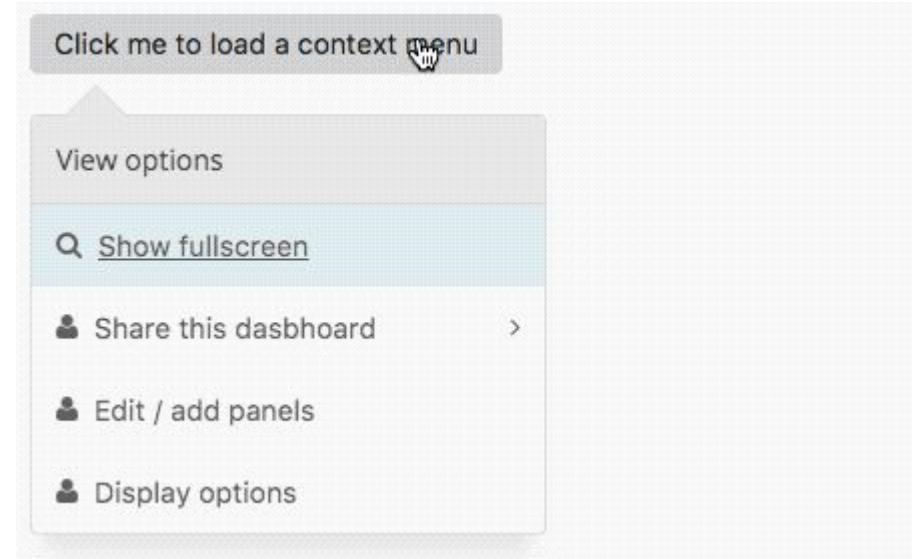
User should be able to navigate within the page without being trapped (e.g. inside a code editor).



The screenshot shows the homepage of the Ace code editor. At the top, there's a navigation bar with links like "Home", "About", "Features", "API", "Downloads", and "Issues". Below the navigation is a large heading "Built for Code". Underneath the heading is a paragraph about Ace's features and its history as the successor to Mozilla Skywriter (Bespin). A code editor window is visible, showing some sample code with line numbers 1, 2, and 3. At the bottom of the page, there's a call-to-action button labeled "Get Ace" and a link to a "kitchen sink" demo.

unless you need to trap focus

In modal elements, e.g. dialogs, contextmenus, usually everything, that closes as soon as you click outside of it.



The screenshot shows a context menu that has been triggered by clicking on a specific element. The menu is titled "Click me to load a context menu" and includes several options: "View options", "Show fullscreen", "Share this dashboard", "Edit / add panels", and "Display options". Each option is preceded by a small icon. The menu is styled with a light gray background and a white header bar. The overall appearance is that of a standard OS X-style context menu.

Make focus ring visible!

(no outline : none)

<LAB TIME>

<https://github.com/tanjadev/keyboard-a11y>

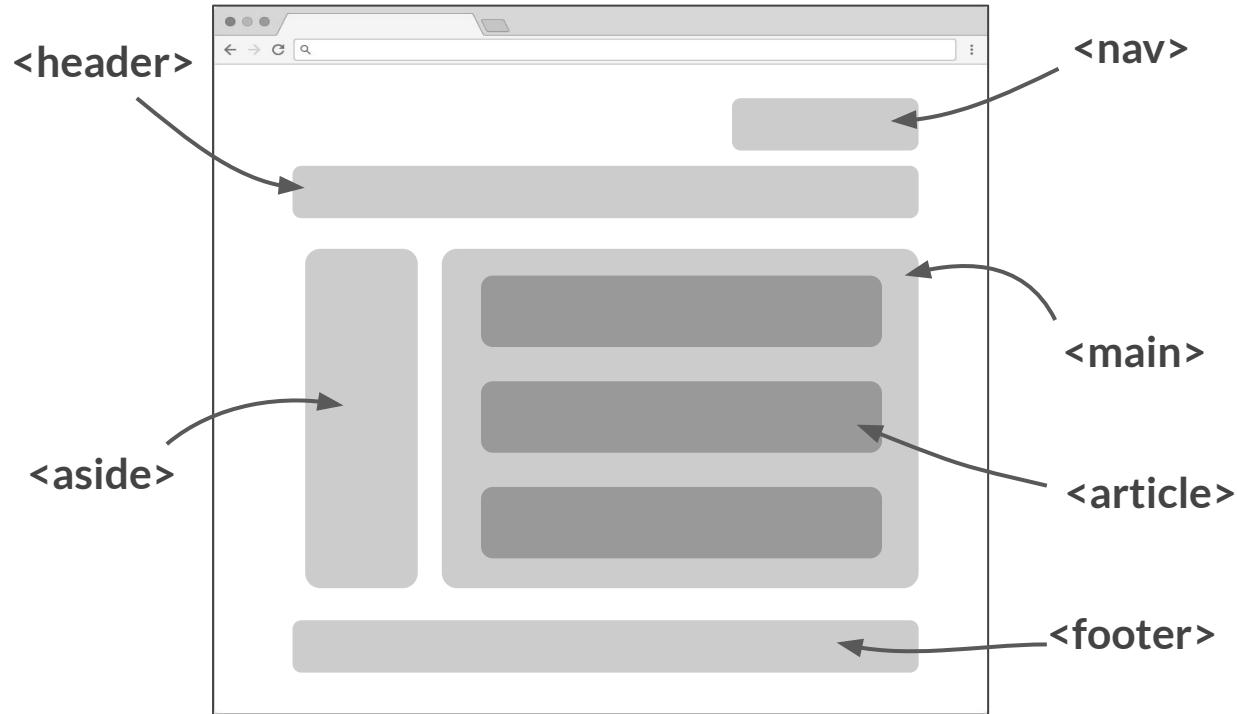


<https://bit.ly/3NPUT2d>

Code Pen: <https://bit.ly/3NRSK6a>

Using HTML:

Semantic HTML



Define Landmarks

```
<aside aria-label="Related W3C Documents">
```

Landmarks

banner

Principles HTML5 Banner Complementary Contenti...

Main Landmark A main landmark identifies the pri...

Coding Techniques region

Landmarks complementary

Related W3C Documents complementary

Take care of your Document

- Proper Heading Structure: <h1> . . <h6>
- Don't skip levels!
- Max one <h1> per page
- Each Page should have a unique <title>
- Define your <html lang="">

Avoid styling tags

display: table (or smth.

else)

<i>

<table> for layouting

Use HTML to structure your content, leave styling to CSS.

<div>

is a container!

is an *inline* container!

Let's talk about
aria=

TABLE OF CONTENTS

Abstract
Status of This Document
1. Introduction
1.1 Rich Internet Application Accessibility
1.2 Target Audience
1.3 User Agent Support
1.4 Co-Evolution of WAI-ARIA and Host Languages
1.5 Authoring Practices
1.5.1 Authoring Tools
1.5.2 Testing Practices and Tools
1.6 Assistive Technologies
2. Important Terms
3. Conformance
3.1 Non-interference with the Host Language
3.2 All WAI-ARIA in DOM
3.3 Assistive Technology Notifications Communicated to Web Applications
3.4 Conformance Checkers
3.5 Deprecated Requirements
4. Using WAI-ARIA
4.1 WAI-ARIA Roles
4.2 WAI-ARIA States and Properties

Accessible Rich Internet Applications (WAI-ARIA) 1.2



[W3C Candidate Recommendation Draft 08 December 2021](#)

▼ More details about this document

This version:

<https://www.w3.org/TR/2021/CRD-wai-aria-1.2-20211208/>

Latest published version:

<https://www.w3.org/TR/wai-aria-1.2/>

Latest editor's draft:

<https://w3c.github.io/aria>

History:

<https://www.w3.org/standards/history/wai-aria-1.2>

[Commit history](#)

Implementation report:

<https://w3c.github.io/test-results/core-aam-1.2/>

Latest Recommendation:

<https://www.w3.org/TR/wai-aria-1.1/>

Editors:

Joanmarie Diggs ([Igalia, S.L.](#))

James Nurthen ([Adobe](#))

Michael Cooper ([W3C](#))

Former editors:

Shane McCarron (Spec-Ops) (Editor until 2018)

Richard Schwerdtfeger ([Knowability](#)) (Editor until October 2017)

Accessible Rich Internet Applications (WAI-ARIA) 1.2

<https://www.w3.org/TR/wai-aria-1.2/>



WHY
ANOTHER
SPEC?!?!?

Many design patterns do not have a native html equivalent

ARIA fills in these gaps

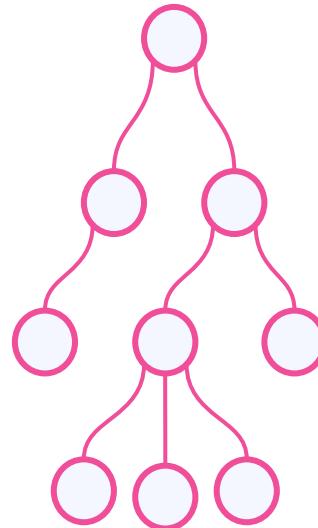
ARIA defines: **roles, states and properties**

Provides context

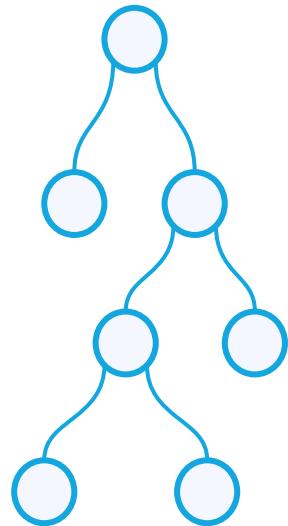
Extends semantics

Where does this information go?

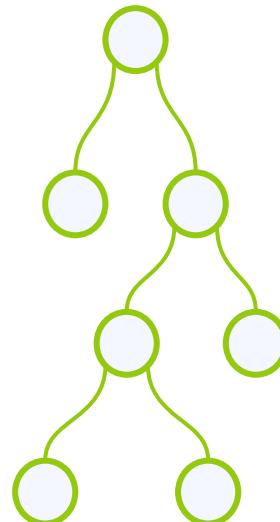
DOM



A11Y Tree



A11Y OS API



Linux: AT-SPI / IAccessible2

Windows: MSAA / IAccessible2 / UI
Automation (UIA)

Mac OS: Ax/uiA

Chromium Dev Tools > Accessibility > Enable full-page a11y tree

Styles Computed Layout Event Listeners DOM Breakpoints Accessibility »

▼ Accessibility Tree

Enable full-page accessibility tree

✓ RootWebArea "WeAreDevelopers World Congress • 14-15 June 2022 • Berlin"

- ✓ button "GET YOUR TICKETS"
 - ✓ StaticText "GET YOUR TICKETS"
 - InlineTextBox

Reload Page and Click On



Elements Console Sources » 0 1 1 2 Switch to DOM

RootWebArea "WeAreDevelopers World Congress • 14-15 June 2022 • Berlin" focusable: true

- banner ""
 - link "WeAreDevelopers" focusable: true
 - StaticText "WORLD CONGRESS"
 - LineBreak ""
 - StaticText "14-15 June • Berlin"
 - LineBreak ""
 - combobox "menu" focusable: true expanded: false
- heading "The World's Largest Developer Event"
 - StaticText "The World's Largest Developer Event"
- heading "WeAreDevelopers World Congress"
 - StaticText "WeAreDevelopers World Congress"
- heading "BUILD.DEPLOY. EMERGE.CONNECT."
 - StaticText "BUILD.DEPLOY. "
 - LineBreak ""
 - StaticText "EMERGE.CONNECT. "
- StaticText "14-15 June 2022 | Berlin"
 - InlineTextBox ""
- LineBreak ""
 - InlineTextBox ""
- button "GET YOUR TICKETS" focusable: true
 - StaticText "GET YOUR TICKETS"

ARIA usage

"If you can use a native HTML element or attribute with the semantics and behavior you require, [...] then do so."

- First Rule of ARIA

Examples:

- Don't use aria-checked, if you can use checked.
- Don't use role="button" if you can use <button>.

Patterns



Read This First

No ARIA is better than Bad ARIA. Before using any ARIA, it is essential to understand why.



Accordion (Sections With Show/Hide Functionality)

An accordion is a vertically stacked set of interactive headings that each contain a title, content snippet, or thumbnail representing a section of content.



Alert

An alert is an element that displays a brief, important message in a way that attracts the user's attention without interrupting the user's task.



Alert and Message Dialogs

An alert dialog is a modal dialog that interrupts the user's workflow to communicate an important message and acquire a response.



Breadcrumb



Button



Carousel (Slide Show)

<https://www.w3.org/WAI/ARIA/apg/practices/>

Providing Extra Description

aria-label="text"

Change the name of that element in the accessibility tree.

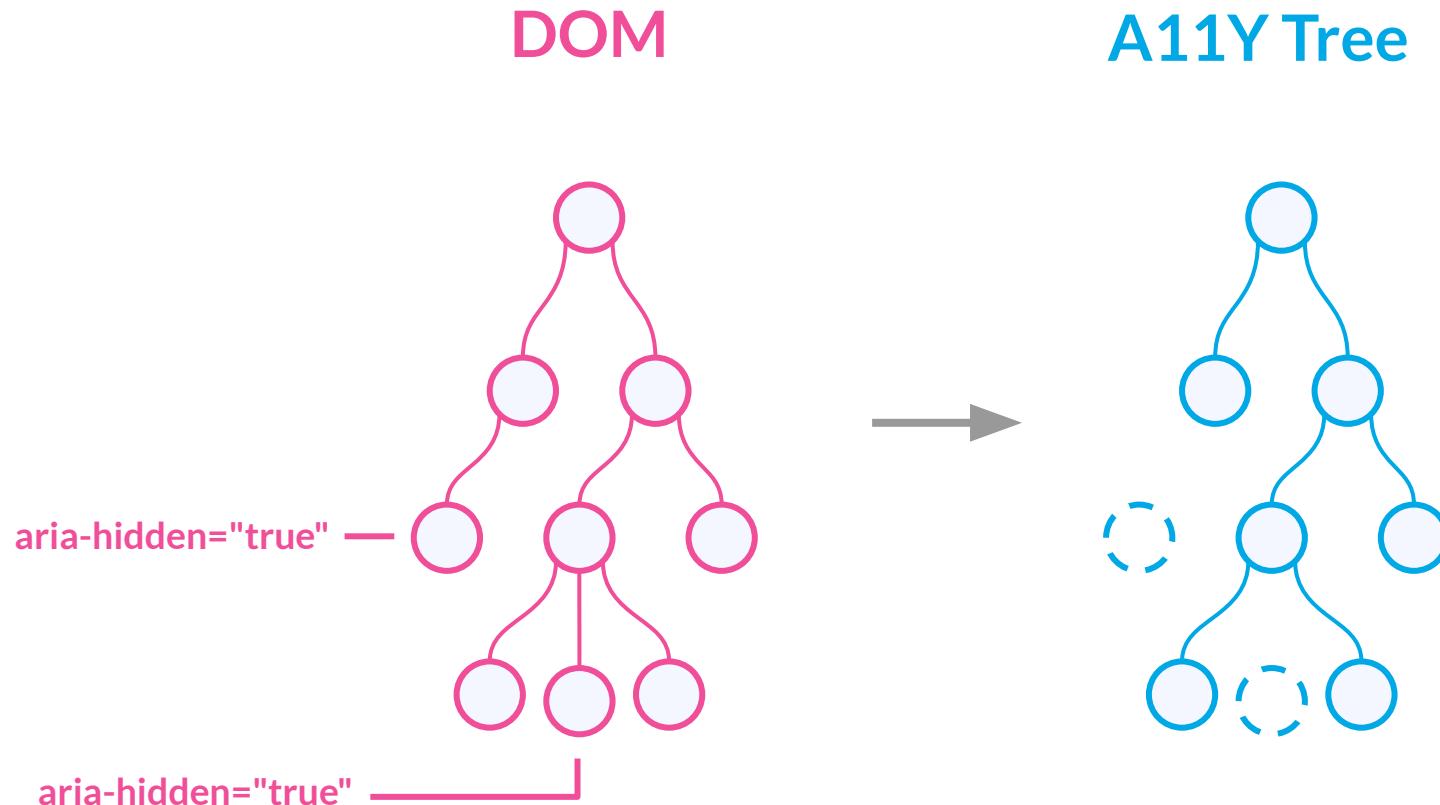
aria-labelledby=<idref>"

Change the name of that element in the accessibility tree to use the content of a references element.

aria-describedby=<idref>"

Add additional information for that element to the content of another element.

Hiding elements from the a11y tree



ARIA example: live regions

```
<div class="chat" role="log" aria-live="polite">  
  ...  
</div>
```

goo.gl/Z59dKQ

No ARIA is better than bad ARIA

ARIA is complex

Browser and Screen Reader Support of specific attributes is inconsistent:

ARIA Browser & Screenreader Support:

<https://a11ysupport.io/>

Always consult the best practices:

<https://www.w3.org/WAI/ARIA/apg/practices/>

Wrapping up

TLDR:

 **a11y benefits everyone**

 **Proper contrast for text and icons**

 **Text alternatives & labels**

 **Tab order is logical**

 **Every action is keyboard accessible**

 **Focus is visible when navigating via keyboard**

 **Use ARIA if you have to**

 **Include automated tests**

Find us, follow us



@inovexgmbh



@inovexlife



inovex



Standorte



Thank you for your attention!

Tanja Ulianova
Fullstack Web Developer

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe

tanja.ulanova@inovex.de



Extra Lab

1. Checkout **github.com/tanjadev/web-a11y-workshop**
2. Use `npm install` and `npm start` to start web application
3. Visit localhost :3000
4. Find "all" accessibility issues (some can be detected automatically, some can't).
5. Make the application properly accessible

If you are stuck: ISSUES.md contains a list of several accessibility issues.