

Accessibility (a11y)

- Content is available to as many people as possible
- Disabilities are common
- Disabilities are more than just blindness
 - But blind people are people too

If humanity and compassion don't motivate you

- Legal requirements
- Technical benefits

Why a11y?

- Programmer are lazy
 - it's one of the 3 Virtues of a Programmer
- a cce ssi bil it y
- a (eleven letters) y
- a11y

Hint: if you put this on your resume, have both forms

Ex: "Exposure to web accessibility(a11y) options"

Warning

- a11y is essential
- It's also hard
 - Most lessons ignore it
 - Easy to be wrong with good intentions
 - Dedication to do well, not afterthought
- Even my govt job wanted to do bare minimum
 - I restrict myself to intro
 - Hopefully solid material

Why You Should Care About a11y

- Caring for people is Good
- Web is ever-more necessary
 - For much of my career that wasn't true
- Legal requirements will hopefully increase
- Demand for a11y-aware devs will increase
 - There's a massive shortage of them

How are we accessible?

- It's an entire field of work
 - We are only covering the intro/basics
- Make HTML inform tools like screen readers
- Provide alternatives for visuals
- Do not rely solely on visual context
- Allow for physical limitations

Informing tools

- Using Semantic HTML
 - Semantics provide automatic behaviors
 - Avoid misleading semantics
- Adding ARIA attributes
 - COMPLEX!
 - Bad ARIA is worse than No ARIA
 - More shortly

Alternatives for visuals

- Image `alt` attributes
 - Have them
 - With **meaningful** text
 - If it is visually interesting, describe it!
 - Even if it isn't mechanically relevant
 - Example: don't say "logo" or "picture"
 - But use `alt=""` for when basically pointless
 - Images in background instead?
 - Don't say "picture of..."
 - just describe contents

Vision isn't a Binary!

- Obey users default text size
 - Use `rem` units for text, not `px`
 - Have most text at `1rem`
- Have text that is large enough
 - Don't shrink font-size below ~12px (`0.75rem`)
- Have sufficient line-height and whitespace
 - You should often raise the default of `1.2`
 - padding between borders and text
- Have enough contrast
 - Text vs background

Don't rely on visuals alone

- Don't use colors/icons alone to signal info!
 - Have text as well
 - Example: an On/Off slider: say "On" or "Off"
 - In addition to any visual effect
 - Don't assume your visuals make sense!

Don't Rely on Visual Context

- Controls/labels that are visually linked
 - Easy mistake to make
- Making a graph accessible?
 - Possible!
 - Before you can research
 - ...you have to even consider it!

Allow for physical limitations

- Allow for keyboard OR mouse
- Minimum size for touch controls
 - At least 24px x 24px
 - 54px x 54px better
- Don't put info needed under their hand (mobile)
- Think before requiring hold/drag/precision
 - Fine motor control isn't even common
 - That's why we call it "fine"
 - Don't require fine motor control

Web Content Accessibility Guidelines (WCAG)

- <https://www.w3.org/TR/WCAG22/>
- WCAG by the WAI at W3C (!)
- A set of guidelines for accessible web content
 - Used by vendors of tools
 - Used by webdevs that care
- 3 levels (A, AA, AAA)
 - A = "must" (absolute minimum, not praise)
 - AA = "should" ("usually good enough")
 - AAA = "may" ("actually working at it")

How to use WCAG

Rules for 4 areas (POUR):

- Perceivable
- Operable
- Understandable
- Robust

Worth it to read through once

- Notably: vague

Semantic HTML covers most of A and AA

- Not everyone is semantic!

Tooling!

Various tools exist to help!

- Tools to test your site
- Tools to act as the user
- Tools to try to do it for you
 - I've heard only bad things

Don't neglect human review!

- Esp. humans that actually know

Why not to rely on validation tools alone!

- Guidelines are vague and subjective
- No tool can test for that
- Tools only recognize clear violations
 - and some might be actually correct
- Human review is needed to find subtle bugs
 - and to verify if reported bugs are real

Why to use validation tools anyway

- Most of us won't know the actual experience
- Good to supplement human review
- Can teach good habits
 - Fix the same issue a few times
 - You start writing it correct the first time

Why to avoid accessibility overlays/edge

- A few companies make these
- Ads/sponsored links in a11y search results
- They offer to make your site accessible
 - Without you changing the site
- These are my personal understanding, not NEU...
 - EVERY a11y expert and disabled user I follow
 - HATES accessibility overlays/edge
 - These tools have lost or settled court cases
- Learn to do it right instead

Example of a validation tool

- aXe, WAVE, etc
- Install WAVE Browser Extension

Example of a screen reader

- (Demonstrate VoiceOver)
- Using a screen reader is good confirmation of UX
 - But involves more work to learn on your part
 - Headsets a must in an office :)
- Demonstrates importance of
 - Semantic landmarks
 - Semantic headings
 - Field labels
 - Image alt text
 - Link/button text

Minimum a11y

- Use Semantic HTML
 - Seriously, not casually
- Provide alt text
- Avoid "Click here" or "Read More"
- Have enough color contrast

Minimum a11y test

(inspired by @geekgalgroks on Twitter)

<https://a11y.jenn.dev/posts/bare-bones-cheatsheet/>

- Can you tab through all controls?
- Can you operate all controls with enter/spacebar?
- Do you pass a color contrast test?
- Confirm alt tags
 - what you tell someone not looking at it

Accessible Rich Internet Applications (ARIA)

- W3C WAI ARIA, for those keeping score
- "Rich" means JS-driven HTML
- HTML attributes to give more meaning
- Semantic elements auto fulfil many of these
- Can be quite complex
- Minimize the need with semantic HTML!

No ARIA is better than Bad ARIA

- ARIA overrides default semantic HTML behavior
 - AND overrides assumptions tools make
 - Because of apps w/o a11y effort
 - When the ARIA is bad, it's a *trusted* bad
 - ARIA assumes behavior, doesn't provide it
- Avoid bad ARIA by minimizing the need for ARIA
 - tired of hearing this yet?
 - and minimize the use of ARIA
 - and understand the use ARIA
 - and verify with screen readers
 - Screen Readers just one assistive tech

ARIA Roles

A "role" gives purpose to an element

- a "button" is a role
- a "heading" is a role

Many semantic HTML elements are roles

- but some people use different elements
- also some roles with no matching element (yet)
 - such as "tab panel" and "tab"

ARIA Landmarks

- Define the foundational structure
 - main
 - navigation
 - region
 - search
 - etc
- You want some, but not too many
 - "noisy"
 - You want to make the page easy to navigate

ARIA States

- States imply changeable states of elements
 - think "checked" or "selected"
 - but also "open", "expanded", etc
- Offer more description than HTML alone
 - and that's when you want a little ARIA

ARIA Properties

Data about an element not expected to change

- such as "label" or "labelled by"

Common use case:

- cards of many articles
- each with intro text and "Read More"
- visually we can see the article title
 - and know "Read more" what?
- ARIA can let us give screen readers more to read

How to ARIA!

- First, do you need to?
- Second, check the Practices document
 - <https://www.w3.org/TR/wai-aria-practices/>
 - Look to see how the ARIA attributes are used
- Third, if you aren't confident, don't use ARIA
 - I've seen criticism of practices doc
 - <https://adrianroselli.com/2019/02/uncanny-a11y.html#APG>

Better Experience through Skiplink

A "skiplink" is a link

- Moves focus past initial headers/navigation
- Benefits users that can't look lower

```
<a href="#main">Skip to content</a>
```

- moves focus to `id=main` element on same page

Nice for everyone when big headers on pages

- Even nicer when it is read to you on every click

Hiding the skip link

Skiplinks are often visually hidden

- not always, but often
- visually shown if you tab to it
- read/usable by screen reader regardless

Only visually hiding the skiplink

`display: none` would REMOVE the skiplink

- meaning it couldn't gain focus
- and wouldn't be read/usable by screen readers

Instead, move away from visual

- transform it offscreen
- still in rendered document
- move it onscreen when it gets focus

<https://css-tricks.com/how-to-create-a-skip-to-content-link/>

Summary - A11y

Accessibility is about making content usable

- Semantic HTML does a lot of work
- Small details can make a big impact
- If it is frustrating for you to fix
 - Consider what it is like for users!

Summary - A11y Tools

- Validation tools are great
 - but cannot be a pass/fail
- Accessibility Overlays exist
 - have a bad reputation
- Screen readers are hard to learn
 - but are "real" experiences

Summary - ARIA

- ARIA are attributes added to elements
 - Provide additional context
 - used by tools to modify experience
- No ARIA is better than Bad ARIA!
- ARIA doesn't create behavior, it informs tools
- Take it slow
 - Minimize use
 - Confirm in tools