ITI0209: User Interfaces

09. Accessibility

Martin Verrev

Spring 2021

When UX doesn't consider ALL users, shouldn't it be known as "Some User Experience" or... SUX?"

Billy Gregory, Senior Accessibility Engineer, Paciello Group.

https://twitter.com/thebillygregory/status/552466012713783297

Accessibility is the practice of making your websites usable by as many people as possible. We usually think of this as being about people with disabilities. But making sites accessible also benefits others - those using mobile devices, or those with slow network connections.

Building accessible sites benefit everyone

- Semantic HTML, which improves accessibility, also improves SEO, making your site more findable.
- Caring about accessibility demonstrates good ethics and morals, which improves your public image.
- Other good practices that improve accessibility also make your site more usable by others - such as mobile phone users or those on low network speed. In fact, everyone can benefit from many such improvements.
- Accessibility is also the law in EU?

Coping with Disabilities (15% world population)

- **Visual Impairments -** blindness, low-level vision, and color blindness. Use screen magnifiers or software zoom. Some users rely on screen readers.
- **Hearing Impairments** - low hearing levels or no hearing at all. No specific equipment exists for web use. There are, however, specific techniques for providing textual alternatives to audio content which range from simple text transcripts to video captions.
- **Mobility Impairments -** such users may not have a mouse The way this usually affects web development work is the requirement that controls be accessible by the keyboard

Coping With Cognitive Impairments

Cognitive impairment refers to a broad range of disabilities, from people with intellectual disabilities to everyone that has sometimes hard time thinking and remembering. A good foundation of accessibility for people with cognitive impairments includes:

- Easily understood content
- Focusing attention on important content.
- Consistent webpage layout and navigation.
- Dividing processes into logical, essential steps with progress indicators.
- Website authentication as easy as possible without compromising security.
- Making forms easy to complete, such as with clear error messages and simple error recovery.

Working Towards Accessibility

- By default, **HTML is accessible**, if used correctly. Web accessibility involves ensuring that content remains accessible, regardless of who and how the web is accessed.
- **Use common sense**. If you are working on a gallery website showing interesting 3D art, it would be unreasonable to expect every piece of art to be perfectly accessible to visually impaired people, given that it is an entirely visual medium.
- **Be realistic**. "100% accessibility" is an unobtainable ideal

Common-Sense Accessibility

Semantic structure

The most important quick win in semantic HTML is to use a structure of headings and paragraphs for your content; this is because screen reader users tend to use the headings of a document as signposts to find the content they need more quickly.

Using native keyboard accessibility

Certain HTML features can be selected using only the keyboard — this is default behavior, available since the early days of the web. The elements that have this capability are the common ones that allow user to interact with web pages, namely links, `<button>` s, and form elements like `<input>`.

Common-Sense Accessibility

Text Alternatives

Text alternatives are very important for accessibility. The simplest text alternative available is the humble `alt` attribute, which we should include on all images. This should contain a description of the image that successfully conveys its meaning and content on the page, to be picked up by a screenreader and read out to the user.

Element relationships and context

There are certain features and best practices in HTML designed to provide context and relationships between elements where none otherwise exists. The three most common examples are links, form labels, and data tables.

Common-Sense Accessibility

Color and color contrast

When choosing a color scheme for your website, you should make sure that the text (foreground) color contrasts well with the background color. Your design might look cool, but it is no good if people with visual impairments like color blindness can't read your content.

Simple functionality

Generally simple functionality should work with just the HTML in place — JavaScript should only be used to enhance functionality, not build it in entirely. Good uses of JavaScript include client-side form validation or custom controls for HTML5 `<video>`.

Common-Sense Accessibility

Using clear language

In general, you should use clear language that is not overly complex and doesn't use unnecessary jargon or slang terms. This not only benefits people with cognitive or other disabilities; it benefits everyone!

- Don't use dashes if you can avoid it. Instead of writing 5–7, write 5 to 7.
- Expand abbreviations — instead of writing Jan, write January.
- Expand acronyms, at least once or twice. Instead of writing HTML in the first instance, write Hypertext Markup Language.

Common-Sense Accessibility

Meaningful Text Labels

UI control text labels are very useful to all users, but getting them right is particularly important to users with disabilities.

You should make sure that your button and link text labels are understandable and distinctive. Don't just use "Click here" for your labels, as screen reader users sometimes get up a list of buttons and form controls.

Make sure your labels make sense out of context, read on their own, as well as in the context of the paragraph they are in.

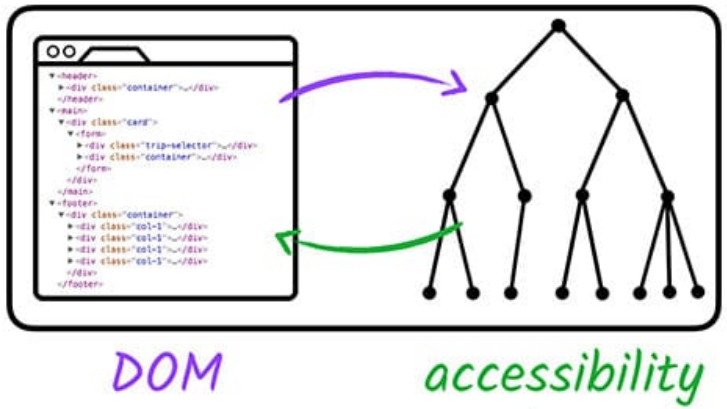
Extra Considerations for Mobile

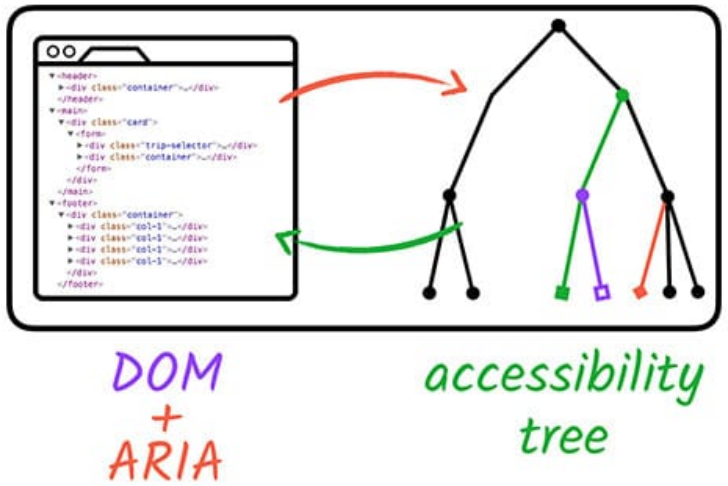
There are some exceptions that need special consideration for mobile; the main ones are:

- **Control mechanisms** — Make sure interface controls such as buttons are accessible on mobiles (i.e., mainly touchscreen), as well as desktops/laptops (mainly mouse/keyboard).
- **User input** — Make user input requirements as painless as possible on mobile (e.g., in forms, keep typing to a minimum).
- **Responsive design** — Make sure layouts work on mobile, conserve image download sizes, and think about the provision of images for high-resolution screens.

Going Further: WAI-ARIA

WAI-ARIA (Web Accessibility Initiative - Accessible Rich Internet Applications) is a specification written by the W3C, defining a set of additional HTML attributes that can be applied to elements to provide additional semantics and improve accessibility wherever it is lacking.





Although ARIA allows us to modify the accessibility tree for any element on the page, that is the only thing it changes. ARIA **doesn't change** any of the element's inherent behavior.

Features WAI-ARIA

- **Roles -** Define what an element is or does. any of these are so-called landmark roles, which largely duplicate the semantic value of HTML5 structural elements e.g. `role="navigation"`
- **Properties -** These define properties of elements, which can be used to give them extra meaning or semantics. For example, `aria-required="true"` specifies that a form input needs to be filled in order to be valid. Properties do not change throughout the lifecycle of an app.
- **States —** Special properties that define the current conditions of elements, such as `aria-disabled="true"`, which specifies to a screenreader that a form input is currently disabled. States can change, generally programmatically via JavaScript.

When To Use

Only use when necessary, you should *always* use native HTML features. See: https://www.w3.org/TR/using-aria/#rule1

1. **Signposts/Landmarks:** ARIA's role attribute values act as landmarks that either replicate the semantics of HTML5 elements (e.g. `<nav>`), or go beyond to provide signposts to different functional areas, e.g search, tabgroup, tab, listbox, etc.
2. **Dynamic content updates:** We can use `aria-live` to inform screenreader users when an area of content is updated via AJAX.
3. **Enhancing keyboard accessibility:** for JS generated elements (using `tabindex`).
4. **Accessibility of non-semantic controls:** for instance `role=button` for JS generated non-semantic controls.

The Web Content Accessibility Guidelines (WCAG) are part of a series of web accessibility guidelines published by the Web Accessibility Initiative (WAI) of the World Wide Web Consortium

Levels of WAI Guidelines

A: Essential

If this isn't met, assistive technology may not be able to read, understand, or fully operate the page or view.

AA: Ideal Support

Required for multiple government and public body websites.

AAA: Specialized Support

Typically reserved for parts of websites and web apps that serve a specialized audience.

Validating Accessibility

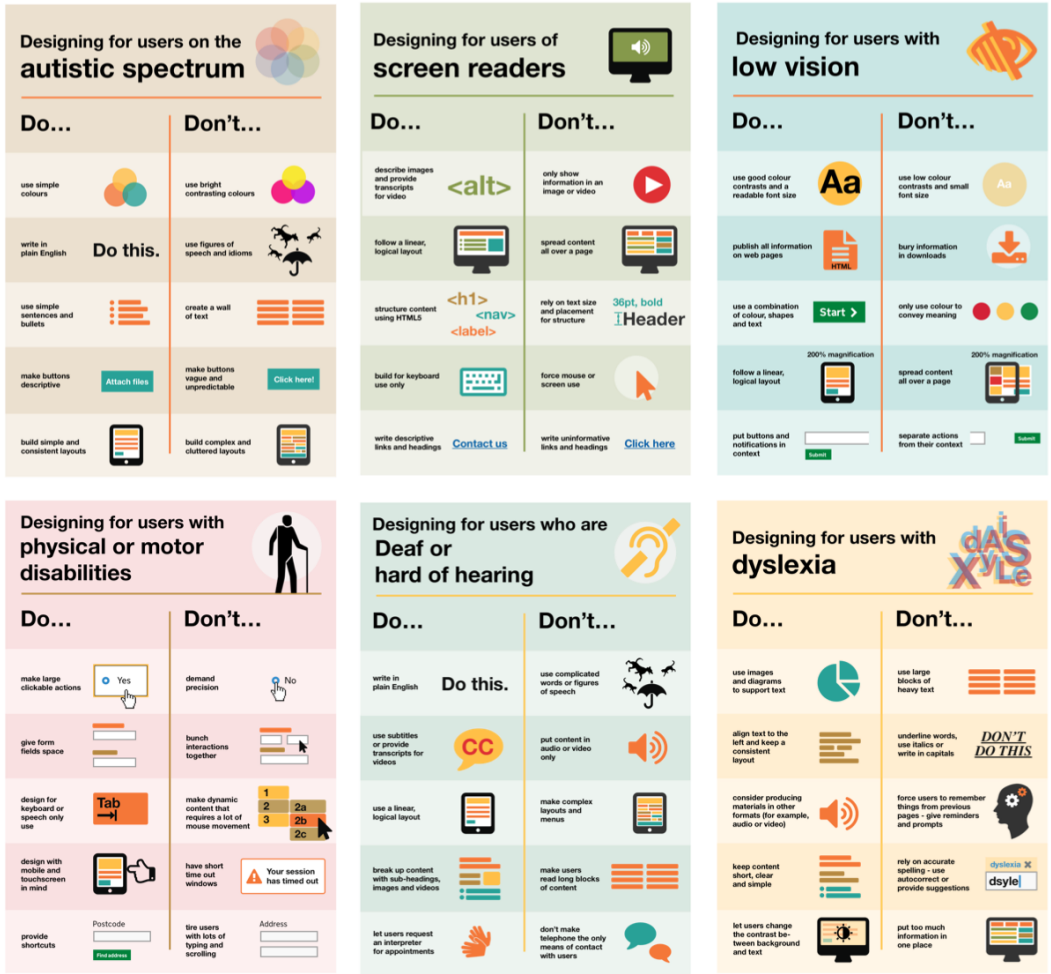
https://wave.webaim.org/

Accessible Rich Internet Applications (ARIA) is a set of attributes that define ways to make web content and web applications (especially those developed with JavaScript) more accessible to people with disabilities.

Tools

- **Screen Reader** for your browser.
- **Colorblinding**. Extension that simulates the website as a color vision impaired person would see.
- **Web Developer Toolbar**. Universal developer utilities and checks.
- **Color Contrast Checker**. https://webaim.org/resources/contrastchecker/ or https://coolors.co/contrast-checker/112a46-acc8e5

A Quick Summary



https://www.techhelp.ca/wp-content/uploads/2021/02/web-accessibility-infographic.png

Links

- What is Accessibility. https://developer.mozilla.org/en-US/docs/Learn/Accessibility/What_is_accessibility
- Web Content Accessibility Guidelines (WCAG) Overview. https://www.w3.org/WAI/standards-guidelines/wcag/
- A11Y Checklist. https://www.a11yproject.com/checklist/
- Bootstrap Accessibility. https://getbootstrap.com/docs/5.0/getting-started/accessibility/
- ARIA on MDN https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA
- Juurdepääsetavus. https://harno.ee/juurdepaasetavus
  Assessment: Accessibility troubleshootin

Thank you!