This checklist uses The Web Content Accessibility Guidelines (WCAG) (https://www.w3.org/WAI/standards-guidelines/wcag/) as a reference point. The WCAG is a shared standard for web content accessibility for individuals, organizations, and governments.

There are three levels of accessibility compliance in the WCAG, which reflect the priority of support:

A

A: Essential

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

AA

AA: Ideal Support

Required for multiple government and public body websites (https://www.w3.org/WAI/policies/). The A11Y Project strives for AA compliance.

AAA

AAA: Specialized Support

This is typically reserved for parts of websites and web apps that serve a specialized audience.

This checklist targets many, but not all level A and AA concerns. Note that the different levels of WCAG support do not necessarily indicate an increased level of difficulty to implement.

Success criteria

Each item on this checklist has a corresponding WCAG "success criterion." Success criterion are the specific, testable rules that power the WCAG, described by a reference number and short title. For example, the rule about text resizing is called 1.4.4 Resize text (https://www.w3.org/WAI/WCAG22/Understanding/resize-text.html).

Some accessibility issues may have multiple success criterion apply to them. We have identified the one most relevant for each checklist item.

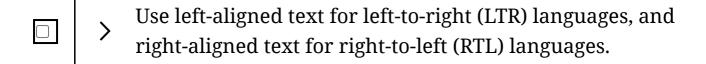# Does this checklist guarantee my site is accessible?

No. However, addressing the issues called out in this checklist will help improve the experience for everyone who uses your site.

The issues this checklist prompts you to check for covers a wide range of disability conditions. There is no such thing as "perfect accessibility" or a site being "100% accessible." You should be wary of companies and services that make such promises. If you need professional accessibility help, use professional accessibility services (/resources/#professional-help).

Content

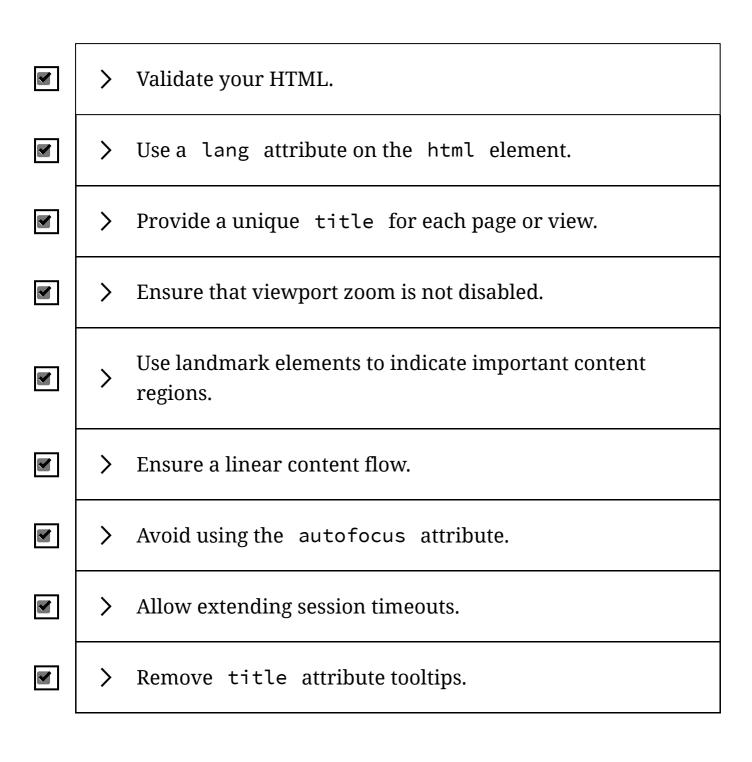Content is the most important part of your site.

☐	›	Use plain language and avoid figures of speech, idioms, and complicated metaphors.
☑	›	Make sure that `button`, `a`, and `label` element content is unique and descriptive.

| ☐ | › | Use left-aligned text for left-to-right (LTR) languages, and right-aligned text for right-to-left (RTL) languages. |

Global code

Global code is code that affects your entire website or web app.

☑	›	Validate your HTML.
☑	›	Use a `lang` attribute on the `html` element.
☑	›	Provide a unique `title` for each page or view.
☑	›	Ensure that viewport zoom is not disabled.
☑	›	Use landmark elements to indicate important content regions.
☑	›	Ensure a linear content flow.
☑	›	Avoid using the `autofocus` attribute.
☑	›	Allow extending session timeouts.
☑	›	Remove `title` attribute tooltips.

Keyboard

It is important that your interface and content can be operated, and navigated by use of a keyboard. Some people cannot use a mouse, or may be using other assistive technologies that may not allow for hovering or precise clicking.

☐	❯	Make sure there is a visible focus style for interactive elements that are navigated to via keyboard input.
☐	❯	Check to see that keyboard focus order matches the visual layout.
☑	❯	Remove invisible focusable elements.

Images

Images are a very common part of most websites. Help make sure they can be enjoyed by all.

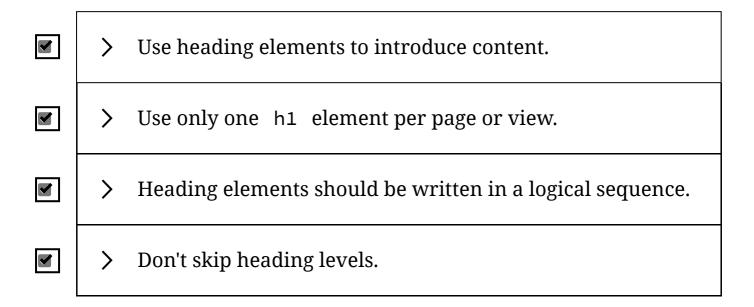|		
☑	❯	Make sure that all `img` elements have an `alt` attribute.
☑	❯	Make sure that decorative images use null `alt` (empty) attribute values.
☑	❯	Provide a text alternative for complex images such as charts, graphs, and maps.

| ☑ | ⟩ | For images containing text, make sure the alt description includes the image's text. |

Headings

Heading elements (h1, h2, h3, etc.) help break up the content of the page into related "chunks" of information. They are incredibly important for helping people who use assistive technology to understand the meaning of a page or view.

☑	⟩	Use heading elements to introduce content.
☑	⟩	Use only one `h1` element per page or view.
☑	⟩	Heading elements should be written in a logical sequence.
☑	⟩	Don't skip heading levels.

Lists

Lists elements let people know a collection of items are related and if they are sequential, and how many items are present in the list grouping.

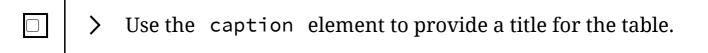| ☑ | ⟩ | Use list elements (`ol` , `ul` , and `dl` elements) for list content. |

Controls

Controls are interactive elements such as links and buttons that let a person navigate to a destination or perform an action.

- [x] › Use the `a` element for links.
- [ ] › Ensure that links are recognizable as links.
- [ ] › Ensure that controls have `:focus` states.
- [x] › Use the `button` element for buttons.
- [ ] › Provide a skip link and make sure that it is visible when focused.
- [ ] › Identify links that open in a new tab or window.
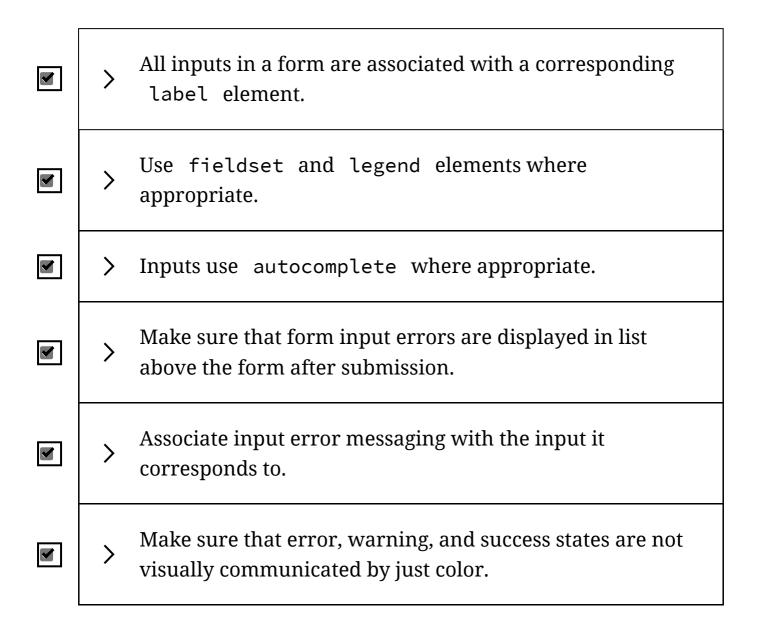
Tables

Tables are a structured set of data that help people understand the relationships between different types of information.

- [x] › Use the `table` element to describe tabular data.
- [x] › Use the `th` element for table headers (with appropriate `scope` attributes).

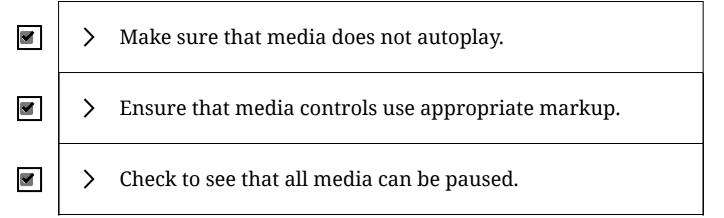| ☐ | ❯ | Use the `caption` element to provide a title for the table. |

Forms

Forms allow people to enter information into a site for processing and manipulation. This includes things like sending messages and placing orders.
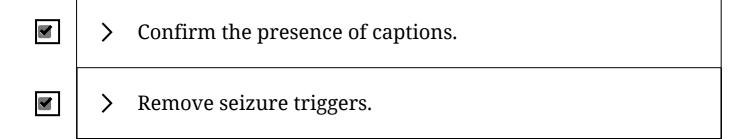
☑	❯	All inputs in a form are associated with a corresponding `label` element.
☑	❯	Use `fieldset` and `legend` elements where appropriate.
☑	❯	Inputs use `autocomplete` where appropriate.
☑	❯	Make sure that form input errors are displayed in list above the form after submission.
☑	❯	Associate input error messaging with the input it corresponds to.
☑	❯	Make sure that error, warning, and success states are not visually communicated by just color.

Media

Media includes content such as pre-recorded and live audio and video.

- ☑ > Make sure that media does not autoplay.
- ☑ > Ensure that media controls use appropriate markup.
- ☑ > Check to see that all media can be paused.

Video

Video-specific checks.

- ☑ > Confirm the presence of captions.
- ☑ > Remove seizure triggers.

Audio

Audio-specific checks.

- ☑ > Confirm that transcripts are available.

Appearance

How your website app content looks in any given situation.

- ☑ > Check your content in specialized browsing modes.

☐	⟩	Increase text size to 200%.
☐	⟩	Double-check that good proximity between content is maintained.
☑	⟩	Make sure color isn't the only way information is conveyed.
☐	⟩	Make sure instructions are not visual or audio-only.
☑	⟩	Use a simple, straightforward, and consistent layout.

Animation

Content that moves, either on its own, or when triggered by a person activating a control.
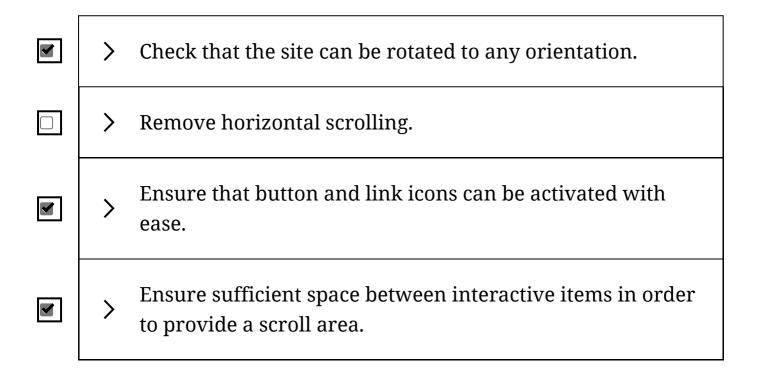
☑	⟩	Ensure animations are subtle and do not flash too much.
☑	⟩	Provide a mechanism to pause background video.
☑	⟩	Make sure all animation obeys the `prefers-reduced-motion` media query.

Color contrast

Color contrast (/posts/what-is-color-contrast/) is how legible colors are when placed next to, and on top of each other.

☑	❯	Check the contrast for all normal-sized text.
☑	❯	Check the contrast for all large-sized text.
☑	❯	Check the contrast for all icons.
☑	❯	Check the contrast of borders for input elements (text input, radio buttons, checkboxes, etc.).
☑	❯	Check text that overlaps images or video.
☑	❯	Check custom `::selection` colors.

Mobile and touch

Things to check mobile experiences for.

☑	❯	Check that the site can be rotated to any orientation.
☐	❯	Remove horizontal scrolling.
☑	❯	Ensure that button and link icons can be activated with ease.
☑	❯	Ensure sufficient space between interactive items in order to provide a scroll area.

Next steps

Remember to periodically check your site to ensure it is still accessible. The A11Y Project also strongly encourages you to verify your testing by hiring a professional tester (/resources/#professional-testers).

Further reading

TetraLogical has a good in-depth, yet still high-level explanation of the WCAG (https://tetralogical.com/articles/wcag-primer/). Check it out if you want to learn more about its history and principles.