Accessibility Highlights

This section provides the comprehensive details, technical guidelines, and full context of the accessibility requirements in this toolkit. It's designed to give you a deep understanding of how to implement these changes effectively. If you need a quicker reference, a short, condensed version of this information is also available as slides, in the PowerPoint file.

Accessible Authentication

What is it for?

When people can't use a login page, the harm can be serious. It can take away their right to privacy, security, and independence.

Logins should not rely on hard memory tests or puzzles, as these block users with cognitive needs. Forcing people to pass these checks often locks them out of vital services.

Logins designed with accessibility in mind are safer and easier to use for everyone.

How to support it

To make login pages accessible, developers should follow these steps:

- Don't rely solely on puzzles or memory tests. Always provide an alternative method.
- Support password managers and allow users to paste text. This improves both accessibility and security.
- Never disable autofill by setting autocomplete to 'off'. Forcing people to type complex details causes unnecessary difficulty.
- Offer flexible alternatives like biometrics, one-time codes, and magic links.

Useful links

- Understanding SC 3.3.8: Accessible Authentication (Minimum), from the World Wide Web Consortium (W3C)

- Accessibility in Cybersecurity: A Practical Guide
- The autocomplete attribute and web security (Mozilla Developer Network)

Accessible Links

Note: Examples of poor link text in this section are shown without a link. The one example link outside the 'Useful Links' section is just for demonstration, so you don't need to follow it.

What is it for?

Everyone has used links. Those wonderful clickable bits of text that take you to magical places (a.k.a. other pages).

Have you seen 'click here' links? A visual user will be able to set it in context and understand the place where the link will direct. But a person relying on assistive technology may scan links on the page, and they will be presented with a list of 'click here' links. They won't be able to understand where each link will direct them.

Sometimes people use the uniform resource locator (URL) as is, like this - https://abiharrisonnye.dev/talks/a11y-series-buttons/transcript. Visual users may not mind this too much, just scanning through it. However, a person relying on assistive technology will have to listen to the whole URL being read out to understand where the link will direct them. This can be disorientating as well as annoying.

How to support it

Link text should make it clear where the link takes you, even without any context. For instance, the link above could direct people to the transcript of my talk, A11y Series: Buttons. If it's in a sentence, make sure it makes sense, while also being clear where it goes to.

You should also avoid non-descriptive link text, like here, click here, read more, and more information.

Useful links

- Yale University - Links
- WebAIM - Links and Hypertext
- University of Reading: Accessible Links

Accessible PDFs

This highlight is about how to make PDFs accessible, and why an alternative is preferred, even after making a PDF accessible.

What is it for?

We use PDF (Portable Document Format) to present all kinds of documents like guidelines, contractual agreements, data-heavy reports and even site maps. Ideally, all the content in a PDF should be accessible to all, and there are ways to make it accessible. Here are a few ways:

- PDFs are not tagged by default, so make sure to enable tagging so that they can be read by screen readers.
- Maintain good colour contrast
- Use big enough fonts and a one-column layout (a little favour to the mobile users, but only a little)
- Add bookmarks to help people move around the document, and set the document title and language in the document properties
- Proper use of headers and footers to provide critical context (document title, section title and page numbers)
- Using semantically correct tags for the content (just like we do for an accessible HTML page)
- Most important: test it with a screen reader like VoiceOver, and with Adobe Acrobat Pro or a PDF Accessibility Checker (PAC). Make sure every part of the content is accessible.

How to support it

Avoid PDFs, even the accessible ones! Yes, PDFs are not at all as accessible as other formats. For instance, PDFs don't adapt to different screen sizes, so they are hard to use on mobile. They have many other downsides compared with other formats too. So always challenge the requirement of using a PDF and prefer other formats instead of PDFs wherever it's possible. Here are a few alternatives:

- HTML web pages and web applications are the most feature-rich alternatives and can be fully accessible when built with semantic HTML.
- Word Documents (with Accessibility Features Enabled) as well as Google Docs / Microsoft Office Online.
    - Offers a built-in accessibility checker.

    - More accessible than PDFs when used properly (heading styles, alt text, etc.)
    - Screen reader-friendly with good keyboard support.
- Accessible Slide Presentations (Google Slides, PowerPoint Online)
    - Can be shared as HTML or accessible PowerPoint format instead of PDF exports.
    - Offers a built-in accessibility checker.
    - Allows embedding of live text, images with alt text, and readable slide order.
- ePub (Electronic Publication Format)
    - Supports reflowable text for different screen sizes.
    - Accessible on e-readers and screen readers.
- Markdown
    - Supports semantic structure and has the ability to provide meaningful alt text for images.
    - Keyboard/screen reader-friendly

Useful links

- Reading PDFs with reflow and accessibility features
- Adobe: Creating accessible PDFs

Accessibility Debt

What is it for?

Accessibility debt is like technical debt, but for accessibility. It refers to all the accessibility issues that build up over time when inclusive practices aren't followed from the start. Just like tech debt, the longer it's left unmanaged, the harder (and more expensive) it becomes to fix.

How to support it

You don't choose to take on accessibility debt. It builds up when things like semantic HTML, alt text, proper labels, or keyboard navigation are skipped or broken.

Here's how to manage or 'pay it off':

- Run regular accessibility audits (manual and automated)
- Add accessibility issues to your backlog and track them like bugs
- Prioritise fixes that block user journeys (e.g. login, forms, navigation)
- Fix inaccessible components in your design system
- Include accessibility in your development and design workflow to prevent new debt

Useful links

- Understanding Accessibility Debt: What It Is and How to Manage It
- Accessibility Debt Is Real — Here's How to Prevent It
- Accessibility Debt: What Happens When You Do Accessibility Last

Accessibility Overlays

What is it for?

Accessibility overlays are automated tools that claim to find and fix accessibility issues on websites. They work by modifying a website with a bit of JavaScript, and usually have an associated toolbar, app, or widget. The aim is that businesses won't have to work on the accessibility of their site, and instead can simply add a pre-built overlay to fix things for them. They can detect some basic issues, such as colour contrast and text size (which isn't enough for a fully accessible site).

They're tempting as a quick-fix solution for accessibility compliance. It sounds appealingly simple - insert a snippet of code, and your website will be 100% accessible. However, these tools are very limited and do not constitute a full compliance solution. Some overlays even add new accessibility issues of their own, such as buttons that can't be used or no screen reader support.

If you'd like to learn more, the Overlay fact sheet explains it better than any summary could. The Overlay fact sheet explains all the reasons to avoid overlays. It also has quotes from the people overlays are meant to help: people with accessibility requirements.

How to support it

Simply - don't! Ensure that you're building accessibility into your designs and into your code, rather than relying on a 3rd party.

Useful links

- Siteimprove - Accessibility overlays
- Overlay fact sheet

Accessibility Testing

What is it for?

Accessibility Testing is the process of checking a digital product to ensure it can be used by people with disabilities. Its purpose is to find and fix barriers before your users do.

Without testing, you are only guessing if your product works for everyone. This can lead to significant problems:

- People using screen readers or keyboards are blocked from completing tasks
- Users with disabilities face a broken or frustrating product
- Your product may fail to meet legal requirements, many of which refer to the Web Content Accessibility Guidelines (WCAG)
- Problems found after launch are much harder and more expensive to fix

How to support it

A good strategy combines three types of testing:

1. Automated Testing
    - This is software that scans code for common, machine-detectable issues
    - It can be used to quickly find problems like low colour contrast or missing form labels
    - However, it only finds approximately 30% of issues. It can't check if the user experience is logical, for instance
2. Manual Testing
    - This is when a human checks the product against accessibility criteria
    - It can be used to find issues that automation misses. Key checks include keyboard-only navigation (can you Tab to everything?) and screen reader testing (does it make sense when read aloud?)
3. User Testing (with people with disabilities)
    - This is the gold standard. Observing people with a range of disabilities as they try to complete tasks on your product
    - It can be used to find real-world usability barriers that technical checks miss. It's the only way to know if your product is truly usable

Useful links

- Best Practices for Accessibility Testing - A Comprehensive Guide
- The Importance Of Manual Accessibility Testing
- Accessibility Testing: How It Works and Why It's Important

Alt Text

What is it for?

Some people cannot see the images we display. This includes people who are blind or have low vision. It also includes people with slow internet, little data, or low battery.

Using alternative text (alt text) ensures that everyone can understand the image content, even if they cannot see it. However, if an image is purely decorative and adds no extra information, give it an empty alt attribute (alt="") so screen readers skip it.

How to support it

- Write short, descriptive text that conveys the actual meaning and context of the image
- Avoid redundant phrases like ‘image of’, as screen readers already announce that
- Leave the description empty for purely decorative elements so assistive tools can skip them
- Plan for alt text during the design and content creation phases instead of at the end
- Use built-in accessibility checkers in software like Microsoft Office to catch missing descriptions

Useful links

- WebAIM Alternative Text
- Write helpful Alt Text to describe images
- What’s the alternative? How to write good alt text

Accessible Rich Internet Applications (ARIA)

What is it for?

ARIA stands for 'Accessible Rich Internet Applications'. ARIA is a set of roles and attributes that enable the web to be more accessible for users with disabilities. ARIA adds to HTML. It lets common web app interactions be passed to assistive technologies when there is no other way to do this.

There are some rules, set out by World Wide Web Consortium (W3C), about how ARIA should be used. They are as follows:

1. If a native HTML element or attribute already has the meaning and behaviour you need, use it. Don't take a different element and add an ARIA role, state or property to make it accessible.
2. Do not change native semantics unless you really have to.
3. All interactive ARIA controls must be usable with the keyboard.
4. Do not use `role="presentation"` or `aria-hidden="true"` on a focusable element. Using either of these on a focusable element will result in some users focusing on 'nothing'.
5. All interactive elements must have an accessible name (W3C definition).

NOTE: Not using ARIA is always better than using ARIA badly. Bad use of ARIA can completely mess up a website for users of assistive technology. ARIA attributes should only be used as a last resort when semantic elements can't achieve the desired result.

The note below is adapted from MDN Web Docs: ARIA:

> Note: There is a saying, "No ARIA is better than bad ARIA." In WebAIM's survey of over one million home pages, they found that home pages with ARIA present had more detected errors on average than those without ARIA. While ARIA is designed to make web pages more accessible, if used incorrectly, it can do more harm than good.

How to support it

There are many different ARIA attributes. The two most commonly used ones are below.

aria-label & aria-labelledby

- Adds an accessible label to elements that do not have visible text. For example, a search button with only an icon should include an `aria-label` clarifying it.
- Use it on interactive elements, like `buttons`, and on landmarks, dialogs, tables and elements with role="img". Don't use it on elements that have no role, like a plain div or span, or on text elements like p, because screen readers may ignore it there.

aria-hidden

- Hides elements from screen readers without removing them visually.
- This is ideal for decorative or redundant elements.

ARIA roles add meaning to content. This lets assistive technologies (such as screen readers) present the content in the way users expect. These should be avoided when there is semantic HTML that can provide the same meaning (e.g. don't use `role=listitem`, and instead use an `<li>`).

Useful links

- W3C - Rules of ARIA
- mdn web docs - ARIA
- Accessibility essentials every front end developer should know
- List of ARIA attributes
- List of ARIA roles
- What I Wish Someone Told Me When I Was Getting Into ARIA

ASCII Art

What is it for?

ASCII art is the practice of arranging punctuation and symbols to create a picture, like the 'shrug' ¯_(ツ)_/¯ or drawings of animals.

While sighted users see a picture, a screen reader reads the content linearly. It cannot 'see' the image. Instead, the user hears a chaotic string of punctuation names: "Slash, backslash, underscore, vertical bar, colon..." It is noisy and blocks them from getting to your actual content.

Screen readers read many symbols and punctuation marks out by name. Grouping punctuation to make pictures creates a 'wall of noise' that is difficult to skip past.

How to support it

You can still share these visuals, but you should change the format to be inclusive.

- Take a screenshot of the ASCII art you want to share.
- Upload the screenshot rather than pasting the text directly.
- Write a description of the image as alt text (e.g. "A line drawing of a rabbit made of text characters").

Useful links

- How screen readers read special characters
- RNIB - Making social media accessible

Assistive Technology (Beyond Screen Readers)

What is it for?

Assistive Technology (AT) is any tool that helps people with disabilities use a computer. It's not just screen readers! Many users rely on other tools to navigate the web. Examples include:

- Screen Magnifiers, for users with low vision to zoom in on parts of the screen
- Switch Devices, for users with motor impairments to navigate using a single button or puff of air, which acts like hitting 'Tab' and 'Enter'
- Voice Control, for users to speak commands like 'Click Submit' or 'Scroll down'

Bad design (like tiny text, layouts that break when zoomed, or buttons that only work with a mouse) can block these users completely.

How to support it

Make your designs compatible. It's simpler than you think and mostly means following the basics:

- Prioritise keyboard navigation. This is the #1 rule. If you can use your site with only the keyboard (Tab, Shift + Tab, Enter, Space, the arrow keys and Escape), you're supporting switch devices, voice control, and more
- Make sure your layout doesn't break or hide text when a user zooms in
- Use clear, simple labels. 'Submit Form' is much better than just a > icon for voice control users
- If you show info on hover, make sure keyboard users can get that same info (e.g. by tabbing to it)

Useful links

- WebAIM - Introduction to Assistive Technology

Basic Accessibility Testing

What is it for?

Testing makes sure that what gets built actually works for everyone. The goal is to never accidentally ship a beautiful page that's completely broken for someone using a screen reader or a keyboard.

While automated tools are awesome, they aren't magic. They only catch about 30% of accessibility errors. An automated tool can check if a picture has an image description (alt text), but it can't tell you if the description actually makes sense. You still have to use your own judgment to check the rest.

How to support it

You don't need to be an expert to test your site. Try these three quick checks right now.

- Use 'Tab' to move forward, 'Shift + Tab' to go back, 'Enter' or 'Space' to select, the arrow keys inside menus and radio buttons, and 'Escape' to close pop-ups. Check that you can reach every button without a mouse.
- Zoom your browser to 200% using 'Ctrl' or 'Cmd' and '+'. Ensure text doesn't overlap or vanish off the screen.
- Run a free scanner like Lighthouse or Axe DevTools in Chrome. It'll quickly flag basic bugs like poor colour contrast.

Useful links

- World Wide Web Consortium (W3C) - Easy Checks (A first review of Web Accessibility)
- WebAIM - Testing with a Keyboard

Breadcrumbs & Wayfinding

What is it for?

Breadcrumbs are a secondary navigation system that shows a user's current location within a site's hierarchy. Their purpose is wayfinding: helping all users understand where they are, where the page sits in the site's structure, and how to navigate back to parent pages. This is essential for users with cognitive impairments and for screen reader users who need context to orient themselves.

Inaccessible wayfinding creates significant barriers. Users get 'lost' and disoriented, screen reader users lose context, and it increases the cognitive load for everyone. This forces users to rely on the 'Back' button or start their entire journey over.

How to support it

To ensure your breadcrumbs are accessible, build them with semantic HTML.

Wrap them in a <nav> element with an aria-label="breadcrumb". This tells screen readers what this landmark is.

Use an ordered list (<ol>) because breadcrumbs represent a sequential path.

The last item (the current page) should not be a link. Identify it with the aria-current="page" attribute.

Hide visual separators (like > or /) from screen readers by wrapping them in a with aria-hidden="true".

Useful links

- Breadcrumbs for Accessibility: Examples & Best Practices
- GOV.UK Design System: Breadcrumbs
- The Admin Bar: Accessible Breadcrumbs

Buttons

What is it for?

Buttons let people take an action, such as submitting a form, saving information or opening a menu. They are one of the most common interactive elements, so if a button isn't accessible, people can be completely blocked from finishing a task.

Accessible buttons help people who use a keyboard, a screen reader, voice control or a switch device, as well as people with low vision or motor impairments.

How to support it

- Use a button for actions and a link for going to a new page. Screen reader users rely on this difference to know what will happen.
- Use the native HTML `<button>` element. It can be reached with the Tab key and works with both Enter and Space, without any extra code.
- Give every button a clear, visible label that describes the action, such as 'Save and continue' rather than 'Click here'.
- If a button only shows an icon, give it an accessible name (for example with `aria-label`) and hide the decorative icon from screen readers.
- Make sure every button has a clear focus state, so keyboard users can see where they are.
- Text on a button needs a colour contrast ratio of at least 4.5:1 (or 3:1 for large text). If the button's edge is what shows it's a button, the edge needs a contrast ratio of at least 3:1 against the background.
- Make buttons big enough to tap easily, with space between them (see Tap Targets).
- Don't rely on colour alone, for example to show that an action is a warning or can't be undone. Make it clear in the button text.
- Avoid disabled buttons where you can. They are often hard to see, and they don't tell the user what they need to do.
- Where you can, reuse a tested button from your design system or component library rather than building a new one.

Useful links

- GOV.UK Design System: Button

- NZ Government Web Accessibility Guide: Buttons for keyboard, mouse and touch
- World Wide Web Consortium (W3C) WAI: Button examples (Accessible Rich Internet Applications (ARIA) Authoring Practices Guide)

Captions & Transcripts

What is it for?

Captions are timed text that appears with a video to show spoken words, music, and sounds, helping viewers who are deaf or hard of hearing. Transcripts are full-text versions of the audio, not timed, and often used for audio-only content like podcasts. The main difference is that captions are synced to the video, while transcripts are not.

While subtitles (a form of captions) are now being used widely by younger generations, captions and transcripts are incredibly important for users with accessibility needs. They are most important for:

- Users with hearing impairments
- Users with cognitive difficulties
- Users who learn best by reading

How to support it

If your video is on YouTube, they have a helpful guide on how to include captions in your videos.

In most video call tools, like Zoom, Microsoft Teams and Google Meet, you can turn on captions in the meeting controls. They might be hidden under 'More'. If the meeting is recorded, many tools can also give you a transcript after the recording has finished.

Creating transcripts can require a bit more work than captions. One straightforward method is to upload your video to a tool that makes automatic captions or transcripts, like YouTube, Microsoft Teams or Slack (in Slack, you can send it to yourself). It will then take a little time to generate a transcript. You can then copy and paste that into a document of your choosing. Once it's in your document, watch the video back, reading through the transcript. Ensure it's all easy to read, remove any weird punctuation from natural pauses, and then share that transcript document alongside your video.

Useful links

- YouTube: Add subtitles and captions
- How to Create a Transcript: A Step-by-Step Guide (2025)

- World Wide Web Consortium (W3C) WAI: Transcripts
- What are Captions?

Carousels & Sliders

What is it for?

A carousel is a sliding banner that automatically cycles through images or messages on a webpage, like a digital billboard. They are very common on websites, but they cause a lot of accessibility problems.

When content moves on its own, people who read slowly or have difficulty focusing can miss it entirely. People who use special software to read screens aloud often cannot follow the content in the right order. And people who cannot use a mouse frequently have no way to control the carousel at all.

There is also a simpler problem: research shows that most people ignore carousels and never click past the first slide. So anything important hidden in slide two or three is missed by almost everyone.

How to support it

The best solution is often to not use a carousel. Showing content in a simple list or grid means everyone can see it, without any of the accessibility problems.

When a carousel is truly needed, it should never move on its own without the user choosing to advance it. The controls need to be big, clear, and work with a keyboard as well as a mouse. And nothing important should only appear inside a carousel.

Useful links

- Should I Use a Carousel? (shouldiuseacarousel.com)
- World Wide Web Consortium (W3C) Carousel Accessibility Tutorial
- SC 2.2.2: Pause, Stop, Hide (Level A)

Colour Blindness

What is it for?

Colour blindness (or Colour Vision Deficiency) is about more than just seeing in black and white. It affects roughly 1 in 12 men and 1 in 200 women. It changes how people perceive the world, which is a huge issue if colour is the only way information is conveyed. For example, a 'red for error' and 'green for success' status light might look the same to someone with red-green colour blindness.

How to support it

Design content to accommodate the various ways that people see. Here are the main types:

- Red-Green is the most common, and it has two main types:
    - Protanopia - Red light looks dark. Crucially, purple often looks just like blue to these users.
    - Deuteranopia - Green light is hard to detect. This means red and green can look almost identical (often a muddy yellow-brown).
- Blue-Yellow (Tritanopia) is much rarer. It makes it hard to distinguish between blue and green, or yellow and red.
- Monochromacy is very rare. This is seeing no colour at all, only shades of grey.

To make your work accessible, never rely on colour alone. Always add a text label, an icon, or a pattern (like hatching on a chart) so the information is clear regardless of the colour.

Useful links

- Coblis - Color Blindness Simulator
- We are Colorblind
- World Wide Web Consortium (W3C) - Use of Color

Colour-blind Series: Protanopia

What is it for?

Protanopia is a form of colour blindness that affects the ability to see red. People with protanopia cannot distinguish between red and green, and red tones can appear very dark or almost black. It's one of two types of colour blindness that fall under the 'red-green colour blind' umbrella. The other, Deuteranopia, affects green-sensitive cells instead. Both cause difficulty telling red from green, but through different mechanisms.

This matters in digital products, because colour is often used to share important information. A form that highlights errors in red, or a chart using red and green bars side by side, becomes very difficult to interpret. For someone with protanopia, that colour information disappears entirely, and the meaning goes with it.

Protanopia is part of a broader family of colour vision deficiencies, and this Colour-blind Series looks at each of the main types, starting with Protanopia. Each type affects a different part of the colour range, but they all teach the same lesson: never rely on colour alone to share something important.

How to support it

The key principle is that colour should always be paired with another visual cue.

- Designers can add icons, labels, or patterns next to colour to show meaning. They can also use colour blind simulation tools to check designs before they are built.
- Developers can ensure interactive states like errors and warnings are distinguishable by more than just colour (e.g. by using icons or supporting text).
- Content writers can make sure any meaning shown through colour is also stated in words.

Useful links

- Web Content Accessibility Guidelines (WCAG) 1.4.1: Use of Colour
- Colour Blindness Simulator (Coblis)
- Colour and Accessibility (WebAIM)

Colour-blind Series: Deuteranopia

What is it for?

Deuteranopia is a form of colour blindness that affects the ability to see green. Like Protanopia, it is a type of "red-green colour blindness". Protanopia affects the cells that sense red. Deuteranopia affects the cells that sense green. It is actually the more common of the two, affecting about 1% of men. It is very rare in women.

In practice, the experience is similar. Red and green become very difficult to tell apart, making colour-coded information unreliable. A traffic light style status indicator, a heatmap, or a form validation message relying on red and green will all present the same challenge.

Deuteranopia is part of a broader family of colour vision deficiencies, and this Colour-blind Series looks at each of the main types. Each type affects a different part of the colour range, but they all teach the same lesson: never rely on colour alone to share something important.

How to support it

The key principle is that colour should always be paired with another visual cue.

- Designers can add icons, labels, or patterns next to colour to show meaning. They can also use colour blind simulation tools to check designs before they are built.
- Developers can ensure interactive states like errors and warnings are distinguishable by more than just colour (e.g. by using icons or supporting text).
- Content writers can make sure any meaning shown through colour is also stated in words.

Useful links

- Web Content Accessibility Guidelines (WCAG) 1.4.1: Use of Colour
- Colour Blindness Simulator (Coblis)
- Colour and Accessibility (WebAIM)

Colour-blind Series: Tritanopia

What is it for?

Tritanopia is a form of colour blindness that affects the ability to see blue. Protanopia and Deuteranopia are both types of "red-green colour blindness". Tritanopia is different: it affects blue and yellow. People with Tritanopia struggle to distinguish blue from green, and yellow from pink or red. It is significantly rarer than the red-green types, affecting less than 0.01% of the population.

In practice, this means colour choices that feel safe for red-green colour blindness can still cause problems. A design that uses blue and green to tell groups apart can be very hard to understand. So can yellow text on white, or white text on yellow.

Tritanopia is part of a broader family of colour vision deficiencies, and this Colour-blind Series looks at each of the main types. Each type affects a different part of the colour range, but they all teach the same lesson: never rely on colour alone to share something important.

How to support it

The key principle is that colour should always be paired with another visual cue.

- Designers can add icons, labels, or patterns next to colour to show meaning. They can also use colour-blind simulation tools to check designs before they are built.
- Developers can ensure interactive states like errors and warnings are distinguishable by more than just colour (e.g. by using icons or supporting text).
- Content writers can make sure any meaning shown through colour is also stated in words.

Useful links

- Web Content Accessibility Guidelines (WCAG) 1.4.1: Use of Colour
- Colour Blindness Simulator (Coblis)
- Colour and Accessibility (WebAIM)

Colour-blind Series: Anomalous Trichromacy

What is it for?

Earlier entries in this series covered full colour blindness, where one type of colour sensing cell is missing. Anomalous trichromacy is different. All three types of colour sensing cells are present, but one of them does not function as strongly as it should. The result is weakened rather than absent colour vision.

There are three types, each a milder counterpart to the full deficiencies covered earlier in the series:

Protanomaly is a reduced sensitivity to red, making it harder to distinguish red from green or brown.

Deuteranomaly is a reduced sensitivity to green, again causing difficulty with red and green. It is the most common form of colour vision deficiency overall, affecting around 5% of men.

Tritanomaly is a reduced sensitivity to blue, making blue and green, and yellow and red, harder to tell apart. It is much rarer than the other two.

This is milder than full colour blindness, so people with anomalous trichromacy can sometimes manage where others cannot. But they will still struggle with designs that rely on small differences in colour, so the same principles apply.

How to support it

The key principle is that colour should always be paired with another visual cue.

- Designers can add icons, labels, or patterns next to colour to show meaning. They can also use colour-blind simulation tools to check designs before they are built.
- Developers can ensure interactive states like errors and warnings are distinguishable by more than just colour (e.g. by using icons or supporting text).
- Content writers can make sure any meaning shown through colour is also stated in words.

Useful links

- Web Content Accessibility Guidelines (WCAG) 1.4.1: Use of Colour
- Colour Blindness Simulator (Coblis)
- Colour and Accessibility (WebAIM)

Colour-blind Series: The Rarer Types

What is it for?

Earlier entries in this series covered Protanopia, Deuteranopia, and Tritanopia. These are the full loss of red, green, and blue colour vision. But colour blindness has a range, and there are other forms worth knowing about.

Achromatopsia is complete colour blindness, meaning a person sees the world entirely in greyscale. It is very rare. People with it are often sensitive to light and cannot see fine detail clearly.

Achromatopsia is one type of monochromacy. Another type is blue cone monochromacy, where only one type of colour sensing cell works at all. This leaves very little colour vision.

These conditions lead to the same design lesson as the rest of the series. If something is shown through colour alone, some users will miss it completely.

How to support it

The key principle is that colour should always be paired with another visual cue.

- Designers can add icons, labels, or patterns next to colour to show meaning. They can also use colour-blind simulation tools to check designs before they are built.
- Developers can ensure interactive states like errors and warnings are distinguishable by more than just colour (e.g. by using icons or supporting text).
- Content writers can make sure any meaning shown through colour is also stated in words.

Useful links

- Web Content Accessibility Guidelines (WCAG) 1.4.1: Use of Colour
- Colour Blindness Simulator (Coblis)
- Colour and Accessibility (WebAIM)

Colour Contrast

What is it for?

Websites, social media images, and out-of-home advertising should have a good colour contrast between text and background colour. That includes links, icons, buttons and any other information. An example of poor colour contrast is light grey text on a white background.

It is useful for, but not limited to, the following people:

- People with vision impairments
- People with colour vision deficiency (also known as colour blindness)
- Elderly people
- People with cataracts
- People in different lighting situations (e.g. sun glare on a phone screen)

It's also worth noting that anything that is indicated by colour also needs to have a secondary way for it to be distinguished. For example, it's no good saying "use the green button if you agree or the red button if you don't" by itself, as some people may not be able to distinguish between those two colours.

The science behind checking if colours are contrasting enough is complex, but you don't need to understand how contrast is calculated in order to use sufficient colour contrast. Colour contrast refers to the difference in luminance (brightness) between the foreground (text) and background colours. Contrast ratios go from 1:1 (no contrast, such as white on white) to 21:1 (black on white). The higher the number, the bigger the difference.

The general rule is that text and images of text must have a contrast ratio of at least 4.5:1. Large text (18 point / 24px or larger, or 14 point / about 18.66px or larger if bold) can have a contrast ratio of 3:1, although it's generally better to aim for a higher contrast.

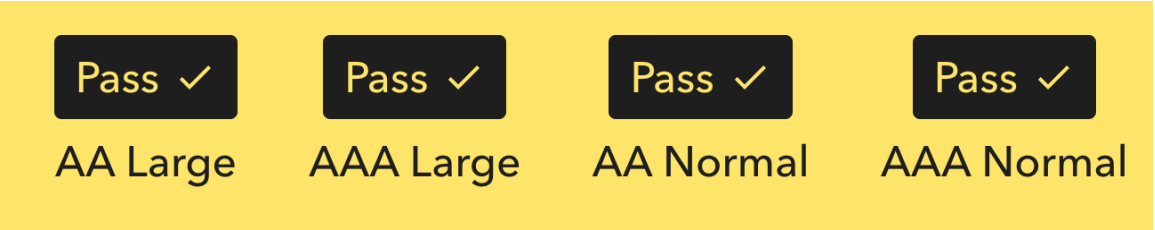
How to support it

Choose colours with sufficient contrast. There are lots of colour contrast checkers online, including:

- Colour Contrast checker (colourcontrast.cc)

- WebAIM's contrast checker
- Coolors contrast checker

To use these, enter your background colour and foreground colour. You want to make sure that it gives you a contrast ratio that is 4.5:1 or higher (so 6:1 is good, 2:1 is bad). It may also show you separate results for AA and AAA, for normal and large text (as shown in the image below). Web Content Accessibility Guidelines (WCAG) AA level is the lower of the two requirements and is the usual minimum to meet. If your text can comply at AAA level, even better.



Useful links

- Colour contrast - Why does it matter?
- Accessible use of colour
- WCAG 2.1 - Contrast (minimum)

Consistency

What is it for?

Think about your favourite supermarket. You know roughly where everything is, and even if you visit a different branch, the layout follows the same pattern. Now imagine if every time you visited, the dairy aisle had moved, the signage used different words, and the tills were in a completely different spot. Even for someone with no disabilities, that would be exhausting and disorienting.

Digital products work the same way. When navigation, buttons, and labels behave consistently across every page, users can build a mental map of the product and move around it confidently. When things shift unexpectedly, that mental map falls apart.

This matters most for people with cognitive or learning disabilities, memory difficulties, ADHD, anxiety or dyslexia. For these users, inconsistency is not just annoying. It can make a product genuinely unusable.

Consistency is something to take into account as part of day-to-day work.

How to support it

Consistency is a shared responsibility across every discipline.

- Designers can ensure the same components are used for the same actions across every screen. If a primary button is yellow on one page, it should be yellow on every page.
- Content writers can use the same labels for the same things throughout a product. Calling something 'Sign in' on one page and 'Log in' on another creates unnecessary confusion.
- Developers can build from a shared component library, so the same element always looks and works the same way. This is better than rebuilding it from scratch each time.

Small inconsistencies that feel trivial to one person can be a significant barrier to another. Think of a menu that moves between pages, a button that changes its label halfway through a journey, or an icon that means different things in different places. Together, these make a product that users can't learn or predict.

Useful links

- Understanding SC 3.2.3: Consistent Navigation (Level AA)
- Making Content Usable for People with Cognitive and Learning Disabilities

Dark Mode

What is it for?

Dark mode is a colour option that flips the user's view from the typical dark text on a light background to light text on a dark background. So instead of blinding white screens, things take on a darker hue. It applies a separate, darker colour scheme to websites, apps, operating systems, and devices to tone down the brightness.

In recent years, the adoption of dark mode has increased, with many users preferring it over light mode across various operating systems.

However, dark mode isn't the all-encompassing accessibility fix that many people believe it to be. Let's take a look at the pros and cons of incorporating dark mode.

Pros

1. Many people prefer dark mode (but not all)
2. On OLED and AMOLED screens, it can be more eco-friendly, as pixels showing pure black are switched off, which saves battery
3. It can enhance readability and reduce eye strain, especially in low-light conditions
4. For users with light sensitivity or photophobia, it provides a gentler visual experience, allowing for better focus

Cons

1. Some visually impaired users can find light mode more comfortable
2. Some users with astigmatism may have issues with reading white text on a black background, as the edges can become blurred. This is similar to the issue caused by street lights when driving at night, and is called halation
3. It can cause issues for users with certain types of colour blindness
4. Lack of consistency across platforms - email is especially complex, which can lead to a bad user experience

How to support it

In code, add a line to use the user's system preferences (using the prefers-color-scheme media query), or add a toggle switch to allow the user to switch between dark and light mode. Ideally, do both: follow the system setting by default, and offer a

toggle. This gives the user full control of their experience. The difficulty is ensuring the correct CSS is in place to handle differing colours.

In emails, it's much more complex, as different email clients will render dark mode differently. Some will use the email's own styling, whereas others apply a basic colour inversion, and others apply their own dark mode styling.

Useful links

- Dark mode & accessibility myth
- Inclusive Dark Mode: Designing Accessible Dark Themes For All Users
- Pros and cons of using dark mode
- 5 myths about dark mode - and the realities

Device Orientation

What is it for?

Think about how you hold your phone. Sometimes you hold it tall (portrait) to read, and sometimes you turn it sideways (landscape) to watch a video.

For some people, turning their device is not an option. A person who uses a power wheelchair might have their tablet bolted to the armrest so it won't drop. If a website forces the screen to be tall, but their tablet is locked sideways, they can't use it. It's like bolting a TV sideways on your wall and being told to just tilt your head to watch it.

How to support it

Websites and apps should work no matter how the screen is turned.

Never use code that forces the screen to stay in just one direction.

Build designs that stretch or stack neatly to fit the screen, whether it is tall or wide.

Useful links

- World Wide Web Consortium (W3C) - Understanding Orientation
- A11y Tip: Device Orientation

Error Messages & Validation

What is it for?

Form errors are a major barrier for all users, but they can completely block users with disabilities. Inaccessible errors include:

- Just highlighting a field in red is useless for screen reader users and people with colour blindness
- Vague messages like 'Invalid input' or 'Error' tell the user nothing about how to fix it
- When a form fails to submit, screen reader users are often left at the top of the page, not knowing what or where the error is

This is a core design pattern, not a single tag. Support depends on using standard HTML and Accessible Rich Internet Applications (ARIA) attributes (`aria-describedby`, `aria-invalid`, `aria-live`) correctly. This is a key part of Web Content Accessibility Guidelines (WCAG)'s 'Perceivable', 'Understandable' and 'Robust' principles and works in all modern browsers.

How to support it

Errors need to be clear, visible, and announced.

- Be specific. Don't say 'Invalid', instead say something like 'Please enter a valid email address (e.g., name@example.com).'
- Don't use colour alone, always add a text message and/or an accessible icon (like an X) next to the field.
- Connect the error. Use `aria-describedby` to programmatically link the input field to its error message. This tells screen readers to announce the error when the user focuses on that field
- Manage Focus. When a form submission fails, ideally, move focus to the first field with an error. If that's not possible, move the focus to a summary of errors at the top of the form
- Use aria-invalid="true". Set this on the input field so assistive tech knows it has an error

Useful links

- World Wide Web Consortium (W3C) WAI - Validating Input

- Level Access - Designer Tips: Creating Accessible Forms

Flashing & Photosensitivity

What is it for?

High-energy videos with strobing effects or rapid transitions can be physically dangerous. For people with epilepsy, if content flashes more than three times per second, it can trigger seizures. For people with vestibular disorders (inner-ear balance issues), intense motion like fast spinning or "shaky cam" can cause instant dizziness, sickness, and vertigo.

How to support it

Safety is about moderation.

- Ensure nothing flashes more than three times in one second
- Avoid rapid spinning or intense zooming
- If a video must contain flashing (e.g. concert footage), add a static "Warning: Flashing images" cover slide so users can scroll past before it auto-plays
- Avoid autoplay on websites; let users choose to start the video manually

A lot of this is manual and relies on people adding warnings and making sure that content is safe for others before it’s shared. However, it's worth noting that social media feeds usually auto-play videos by default. The platform won't stop a dangerous video, so it is entirely up to the person posting to check that their content is safe before sharing.

Useful links

- Epilepsy Society - Photosensitive Epilepsy
- Designing Safer Web Animation For Motion Sensitivity

Focus States

What is it for?

Focus states occur when an interactive element receives focus. Interactive elements include links, buttons, and input fields. You can focus on an element by clicking it (as long as the click doesn't take you somewhere else). You can also press the Tab key until the element is selected.

Focus states have their own requirements in Web Content Accessibility Guidelines (WCAG). The focus indicator must be visible (SC 2.4.7, Level AA). It needs a contrast ratio of at least 3:1 against the colours next to it (SC 1.4.11, Level AA). It must not be completely hidden by sticky headers, footers or other content (SC 2.4.11, Level AA, new in WCAG 2.2).

How to support it

If you're building or designing a new component, make sure its focus state meets all three of these requirements.

Useful links

- Understanding focus state styles
- Wise - Focus states
- Focusing on focus states

Focus Traps

What is it for?

A focus trap keeps keyboard focus inside one part of the page while it's open, such as a modal. Pressing Tab only moves between the items inside it, so the user doesn't get lost on the page behind it. A good focus trap always lets the user leave, for example by pressing Escape or using a close button.

How to support it

Only use a focus trap where it's needed, such as in a modal. Unwanted traps can be added by accident. Always test your pages using only the keyboard before any changes go live.

If you really do want a focus trap, use a ready-made accessible component like a Modal. It handles the focus trap for you.

NOTE: It’s very important to understand that this highlight is about focus traps, NOT keyboard traps. A keyboard trap fails SC 2.1.2: No Keyboard Trap (Level A). That’s when you end up stuck somewhere, such as a modal, and you can’t get out of it without using a mouse!

Useful links

- A Primer on Focus Trapping
- How to Build Accessible Modals with Focus Traps

Fonts

What is it for?

Fonts are used everywhere, whether that's in a Word document, in a design, in a slide deck, or on a website. Not all fonts are created equal. Many people will have come across Wingdings and similar fonts (Wingdings has an interesting history; for more information, see an article on Wingdings font history), so we're all aware of fonts that are completely unreadable. However, some fonts may appear to be easy to read to some, but incredibly challenging for others.

Fonts are impactful to everyone, but especially for the following people:

- People with reading difficulties (like dyslexia)
- People with visual impairments

How to support it

This section focuses on what to look for when choosing a font.

Mirroring is when a reader mixes up two characters, because one looks like the other turned around. For example, someone might read a lowercase b as a d, or 6 as 9. This means the word bog could be read as dog.

Transposition is when letters seem to swap places. For people with dyslexia and other reading difficulties, this is often a bigger problem than mirroring. For example, someone might read “loin” as “lion”, or “lien” as “line”.

To help with mirroring and transposition, choose a font where characters like these look clearly different from each other:

- q p
- d b
- 0 O
- n u
- i l 1 I
- 6 9
- a 8
- a 6
- 6 g 9

- `rn` `m`

When choosing a font, especially a sans serif (Google Fonts glossary), compare the uppercase `I`, lowercase `l` and number `1`. In many fonts, all three are a single straight line, so they're easy to mix up. Look for a font that makes them different, for example a flat base on the `1` or a tail on the `l`. Small differences in height or line thickness also help.

Fonts such as Atkinson Hyperlegible Next have become popular with individuals who have visual or cognitive disabilities.

Useful links

- Introducing accessibility in typography
- Siteimprove: Accessible fonts
- What is an accessible font?

Forms

What is it for?

An accessible form is an online form that everyone can fill in, including people with vision, movement, or cognitive disabilities. It collects information, such as contact details, sign-ups, feedback, or payments. It does this in a way that is clear and easy for everyone to use, including people who use screen readers or only a keyboard.

Inaccessible forms create significant barriers that can prevent people from completing essential tasks. These barriers include:

- Missing or unlinked labels: Screen reader users don't know what information to enter in a form field
- Poor keyboard navigation: Users who cannot use a mouse are unable to 'Tab' to form fields, buttons, or links in a logical order, or they get trapped in one section
- Vague error messages: Errors that only use a red border (relying on colour alone) or say "Invalid input" don't tell a user what is wrong or how to fix it
- Small tap targets: Small radio buttons or checkboxes are difficult to select for users with motor impairments
- Overly complex layouts or time limits can create cognitive barriers, making the form confusing or stressful to complete

All modern web browsers and assistive technologies (like screen readers) fully support the standard HTML elements (`<label>`, `<input>`, `<fieldset>`, `<legend>`, etc.) and Accessible Rich Internet Applications (ARIA) attributes needed to create accessible forms.

How to support it

To ensure your form is accessible, you should build it with semantic HTML and clear instructions from the start.

- Always use `<label>` tags and associate them with their inputs (using the `for` and `id` attributes)
    - Placeholder text (e.g., "Enter your name") is not a substitute for a label
- Group related fields

    - Use a `<fieldset>` to group related items (like a set of radio buttons for a "Yes/No" question, or the fields for a shipping address) and a `<legend>` to describe the group
- Provide clear error validation
    - When a form has errors after the user submits it, the message should be in text, identify the specific field, clearly describe the problem, and suggest a correction
- Ensure full keyboard accessibility
    - A user must be able to navigate to every form field, link, and button using only the keyboard (for example 'Tab', 'Shift + Tab', the arrow keys and 'Space'), in a logical sequence.
- Use high-contrast colours for text, borders, and controls, and ensure all clickable elements have a large enough target size.

Useful links

- World Wide Web Consortium (W3C) WAI: Forms Tutorial
- WebAIM: Creating Accessible Forms
- MDN: Web Forms

Good Alt Text

What is it for?

While basic alt text just labels an image, good alt text explains its purpose. It bridges the gap between seeing a visual and understanding why it is there.

- Delivers the exact message, data, or emotion the image is meant to convey
- Provides an equivalent user experience instead of just ticking a compliance box
- Prevents confusion caused by vague descriptions like ‘chart’ or ‘screenshot’
- Focuses only on the details that matter to the surrounding text

How to support it

Supporting good alt text means moving past basic labels and focusing on the specific message your image is meant to communicate.

- Think about what information a user misses if the image fails to load
- Tailor the text to match the specific context of the surrounding page
- Avoid bad examples like ‘image.png’ or ‘graph’, which give the user zero useful context
- Use good examples like ‘Line graph showing a twenty percent increase in app sign-ups this quarter’ to provide immediate value

Useful links

- Write helpful Alt Text to describe images
- Good Alt Text, Bad Alt Text — Making Your Content Perceivable
- What’s the alternative? How to write good alt text

Head Pointer

What is it for?

Head pointer allows users with limited hand function to control a computer cursor using head movements and, in some cases, facial expressions. It's an accessibility feature available on macOS (sorry, Windows and Linux users) and can be enabled through System Settings.

While Head Pointer is great fun to play with, it's especially important for the following people:

- People with motor skills disabilities
- People who struggle with fine motor control
- People with tremors
- People with little to no use of their hands (or no hands at all)

How to use it

As Head Pointer is so customisable, a full breakdown isn't possible here. However, the short version is that it can control your cursor by tracking your head movements, and it can use facial expressions to perform certain actions. Try sticking your tongue out to left-click, or if you can wink, that's a fun option to experiment with!

Take a look at the links below, get it set up to your preferences, and see if you can survive an hour working with it!

Useful links

- Move the pointer using Head Pointer on macOS
- An introduction to macOS Head Pointer

Headings

What is it for?

Most people are familiar with headings, whether that's a title in a document or a title sheet on a slide deck. But what some people aren't aware of is that headings are important for more than just making an impact.

Headings are especially important for the following:

- People who use screen readers
- People with cognitive difficulties
- Search engines

How to support it

Headings are simple to get right, but are often misunderstood. To start with, headings shouldn't be used for styling. Using a 'Heading 2' in your Word doc for some large, bold text might give you the right font size and font weight (how bold it is), but it also tells a screen reader that your text is a heading. This meaning may not be accurate. The 'meaning' here is called 'semantics'. Semantics are often referred to as 'building blocks' - without them, everything falls apart!

The examples below refer to the code elements of headings and the heading style names in Word and Google Docs. However, the principle applies to everything.

To start with, there should only be one title (or `h1`) per document, slide, or web page.

After that, headings should follow in a sequential order. So a Heading 1 (`h1`) should be followed by a Heading 2 (`h2`). A Heading 2 (`h2`) should be followed by a Heading 3 (`h3`). A Heading 3 (`h3`) should be followed by a Heading 4 (`h4`), and so on. You can return to previous heading levels, but you should never skip a heading level on your way down (so Heading 4 (`h4`) to Heading 2 (`h2`) is perfectly fine, Heading 2 (`h2`) to Heading 4 (`h4`) is not).

You shouldn't use a heading to get a certain font size or styling. Instead, style it yourself. Headings convey meaning to screen readers and search engines, so you should only use headings for text that is necessary as a heading; other text should have the styling changed to match designs.

Search engines are effectively 'blind'. They don't see visually; instead, they scan through the code of what they are given. If you're using headings incorrectly, search engines will get the wrong information, and search engine optimisation (SEO) can suffer as a result.

Useful links

- How to: Accessible heading structure
- Arizona State University: Accessible headings
- World Wide Web Consortium (W3C) WAI: Headings tutorial
- Home Office Design System: Headings

Hover states

What is it for?

Hover states are what happen when a mouse pointer moves over an interactive element. Interactive elements include things like links, buttons, and inputs.

Hover states must also follow all the usual accessibility guidelines. For instance, if extra content appears when someone hovers over or focuses on an element, it must stay visible while they move the pointer onto it (screen magnifier users often need to do this to read it). They must also be able to dismiss it, for example with the Escape key. This is Web Content Accessibility Guidelines (WCAG) SC 1.4.13 Content on Hover or Focus (Level AA).

How to support it

If you're building or designing a new component, make sure it has hover states that follow all accessibility guidelines (e.g. colour contrast, target size, descriptive text, etc.)

Useful links

- A whole lot of bovver over hover
- Accessible button design
- Hover Actions and Accessibility: Addressing a Common WCAG Violation

Inclusive Language

What is it for?

Inclusive language means using words that respect everyone. It avoids words that leave out, sideline, or put down groups of people, for example because of disability, race, gender, sexuality, age, or wealth.

In accessibility, language is a critical part of the user experience. Non-inclusive language creates significant barriers, making people feel unwelcome, unseen, or disrespected. These barriers include:

- Ableist language (e.g., 'lame', 'crazy', 'blind spot') uses disability as a negative metaphor, which can be offensive and alienating
- Biased or gendered terms (e.g., 'salesman', 'guys') can exclude people who don't identify with those terms
- Complicated jargon or sayings can be real barriers for people with cognitive disabilities, people who don't speak English as a first language, or people who find reading hard
- Using labels (e.g., 'the disabled') defines people by their condition rather than as individuals

This is a communication practice, not a technical feature. It is fully supported and encouraged across all platforms and forms of communication.

How to support it

To ensure your language is inclusive, you must be conscious, precise, and empathetic in your word choices.

Use person-first language (e.g., 'a person with vision loss') unless a group or individual expresses a preference for identity-first language (e.g., 'an autistic person'). When in doubt, person-first is often a safer starting point. The Deaf community generally prefers identity-first language (e.g., 'Deaf people').

Avoid ableist metaphors. Instead of 'that idea is lame', say 'that idea is underdeveloped'. Instead of 'fell on deaf ears', say 'the feedback was ignored'.

Use neutral terms. Opt for 'everyone', 'team', or 'you all' instead of 'guys'. Use 'parent' instead of 'mother/father', and 'partner' instead of 'husband/wife/boyfriend/girlfriend'

if you don't know.

Be specific and clear. Avoid jargon and write in plain language. Clearly describe functionality without assuming ability (e.g., instead of 'Click here', use 'Select the 'Submit' button').

Useful links

- Home Office Design System: Inclusive language
- Harder than you think: developing an inclusive language guide
- The Quick Guide to Inclusive Language

Keyboard Only

What is it for?

Keyboard only is a method for navigating the web using just a keyboard (not using the mouse at all). Many people with disabilities navigate using only the keyboard, but some users without disabilities also navigate using the keyboard for speed and efficiency.

Many people navigate using their keyboard only, but it's especially important for the following people:

- People with motor skills disabilities
- People who struggle with fine motor control
- People with tremors
- People with little to no use of their hands (or no hands at all)
- People with visual impairments

How to support it

Make sure that all interactive elements (buttons, links, etc.) can be navigated to using the keyboard, and that all interactivity still functions as expected. The focus should also be clear: if your focus is on a button, can you tell that's where your focus is just by looking?

To test keyboard accessibility:

1. On page load, don't click anywhere. If you've already clicked, click in the top left corner of the page
2. Press `Tab` to tab through the page
3. `Shift` + `Tab` will tab backward
4. Arrow keys move within radio groups, menus and tabs
5. `Enter` activates links and buttons
6. By default, `Space` activates buttons and checkboxes. When nothing that takes Space input is focused, it scrolls the page

Test keyboard accessibility on mobile devices as well, as users with disabilities often use an external keyboard with phones and tablets.

Useful links

- WebAIM: Keyboard accessibility
- MDN: Keyboard accessible
- How to Enhance Accessibility with Keyboard-only Navigation

Landmarks & Roles

What is it for?

Imagine trying to navigate a city with no street signs, where every building looks exactly the same. That is what a website feels like to a screen reader user without Landmarks and Roles.

- Landmarks: These are like zones on a map. They tell the user 'This is the Navigation' or 'This is the Main Content'. They let users jump right to what they want to read.
- Roles: These are like name tags. They tell the user exactly what an item is. A plain box tells the user nothing, but a 'Button' role says, 'Click me!'

It is supported universally. All browsers understand standard HTML tags automatically. It's just a matter of choosing the right ones.

How to support it

The best way is to use semantic HTML (standard HTML tags). When you use the right tag, you get the Landmark or Role for free.

- Landmarks: Use <nav> for menus and <main> for your main content.
- Roles: Use <button> for buttons and <a> for links.

The Catch

You can use code to force a plain box (`<div>`) to act like a `<button>`, but it is risky, and takes much more work.

- Using a `<button>` tag is like buying a chair. It comes ready to sit on.
- Using a `<div>` is like buying a pile of wood and writing 'Chair' on it. It says it's a chair, but you have to build the legs yourself (e.g. add keyboard support) or it will fall apart.

Useful links

- CSS Tricks - Why, How, and When to Use Semantic HTML and Accessible Rich Internet Applications (ARIA)
- HTML Semantics and Accessibility Cheat Sheet

- How to Improve Web Accessibility with Landmarks

Link Text

What is it for?

Many screen reader users navigate websites by pulling up a list of only the links on a page. If your links just say 'Click here' or 'Read more', their list will just read out, 'Click here, Click here, Read more'. That doesn't help them find what they are looking for. Good link text makes sense all by itself, even if you hide the rest of the page.

Good link text doesn't need any special code. It's all about writing clear, helpful words, and everyone benefits from that, not just screen reader users.

How to support it

Write links that tell the user exactly where they are going, what they're doing, or what they are opening

- Be descriptive
    - An example of non-descriptive link text - 'To see our new report, click here'
    - An example of descriptive link text - 'Read our 2024 Report'
- Avoid raw uniform resource locators (URLs). You should never use a long web address as the link text (like `https://www...`). Screen readers will read out the whole address, including slashes and symbols, which is very frustrating to listen to
- Be clear and brief. You want to keep link text short - tell them the destination without writing a whole paragraph

You might think that the sentence right next to the link explains what it does. The catch is that users jumping from link to link do not hear the surrounding text. They only hear the words that you made clickable

Useful links

- Writing Meaningful Link Text
- WebAIM: Links and Hypertext
- World Wide Web Consortium (W3C): Link Purpose

Meetings

What is it for?

Meetings are part of most workdays, and there are many ways to make them fully accessible and inclusive. This helps everyone, but it can be especially impactful for:

- Neurodivergent people
- People with anxiety
- People with learning disabilities

How to support it

Start with an inclusive meeting invite, such as this inclusive meeting invite template. Ask attendees whether they have any accessibility requirements, and let them know which accessibility features are available during the meeting. Give all the details about the meeting up front. This gives people time to prepare, read anything they need to, and understand what is being asked of them. Say whether it's a remote, hybrid, or in-person meeting, and whether the meeting will be recorded or not.

Some other things you can consider are:

- Give a short visual description of yourself for anyone who might be visually impaired. This is especially important for any meetings that will be recorded for future use
- Let everyone know your pronouns (you can also add them to your display name)
- Shorten your meetings by 5-10 minutes, to avoid anyone having no gaps between meetings
- Allow for people to move, doodle, fidget or stand during the meeting
- Let your attendees know that they can turn on closed captions or the meeting transcript (this is often hidden under 'More' in the meeting controls)
- Include breaks for meetings longer than an hour

Useful links

- University of Reading: Accessible meetings
- Top tips for hosting accessible and inclusive online events
- Accessibility tips for inclusive Microsoft Teams meetings and live events

Modals

What is it for?

Modals are manually triggered pop-ups that provide extra information. For example, a shop might use a modal to explain what each condition grade of a refurbished phone means, opened from a link next to the grades.

Modals make it possible to include all the required information on a page, without presenting users with a giant page full of text. It means they can choose when to view more information about certain things. Modals can also be used for filtering and sorting options, allowing those to take up less space too.

Modals are incredibly useful for (but not limited to):

- People with cognitive impairments
- People with ADHD
- People with autism
- Screen reader users

Almost fully supported (Can I use: dialog element) - Internet Explorer, Opera Mini and KaiOS don't support it, though those aren't widely used.

How to support it

In general, modals can be tricky to build accessibly. Important considerations include the user's focus and keyboard navigation.

- When the modal opens, move focus into it
- While it's open, keep focus inside the modal so keyboard users can't tab to the page behind it
- Let users close it with the Escape key as well as a visible close button
- When it closes, return focus to the element that opened it

The native HTML `<dialog>` element handles much of this for you, so use it, or an existing accessible modal component from your design system, rather than building one from scratch.

Useful links

- Accessibility Essentials - Modals

Neurodiversity & Inclusive Design

What is it for?

Neurodiversity describes the natural variations in how human brains function, process information, learn, and interact. This includes conditions like autism, ADHD, dyslexia, and dyspraxia. Under UK law (the Equality Act 2010), neurodivergence can count as a disability if it has a substantial, long-term effect on someone's day-to-day activities. But not all neurodivergent people see themselves as disabled.

Designing for neurodiversity is a way of making websites and apps that are clear, predictable, and easy to use for people with different ways of thinking. Bad design can create problems for thinking and processing, such as:

- Busy layouts, auto-playing videos, and flashing animations can cause cognitive overload, where there's too much information to take in at once
- Using jargon, complex words, or vague phrases means that some people simply can't understand what you're trying to say
- Menus, buttons, or layouts that change from page to page mean that people don't know what to expect, and it takes more cognitive work to understand
- Long blocks of text make it hard to find information
- Unnecessary urgency can cause stress, such as short time limits for submitting a form

This is a way of designing, not a single feature. Key web standards support this approach, especially the ‘Understandable’ and ‘Operable’ parts of the Web Content Accessibility Guidelines (WCAG). It's about using standard HTML and CSS in a thoughtful way to create a better experience for everyone.

How to support it

To make an experience better for everyone, focus on making things clear, predictable, and flexible.

- Keep it simple and clear, use plain English and short sentences
- Choose simple, easy-to-read fonts
- Make it predictable, use headings and lists to organise content. Keep your layout the same on every page, and make sure buttons clearly say what they do (e.g., "Submit your form")

- Reduce distractions, letting the user be in control. Avoid auto-playing videos or flashing animations
- Be forgiving. Write error messages that clearly explain what is wrong and how to fix it
- Make content easy to read by breaking up long text with headings, lists, and empty space

Useful links

- Neurodiversity and designing for difference
- Scott Logic: Designing for Neurodiversity

Personas

What is it for?

Essentially, Personas are fake people. Their lives, issues, and accessibility requirements allow you to role-play as that person while you test for accessibility. This brings a human side to accessibility testing. They aren't real, but real people live with the same conditions and needs.

How to support it

Start by going to the HMRC Personas website, and run through each of their Personas, attempting the tasks that are set. Once you've done that, take the learnings and the methods that you've used, and use them to test your own work for good accessibility practices.

Useful links

- HMRC Personas

Prefers-reduced-motion

What is it for?

Some people struggle with motion on websites. This can be due to a number of reasons, including (but not limited to):

- ADHD
- Motion sickness
- Vestibular disorders

While it's best not to use any motion that autoplays, sometimes it's unavoidable. So that this doesn't create barriers for users, you can use the prefers-reduced-motion media query to stop the motion from happening for those who choose not to see it.

Reducing motion can have other benefits too, such as saving battery life and data on mobile devices.

Many new operating systems allow you to choose to reduce the motion that you see on websites (at least, on websites where they've used the prefers-reduced-motion media query).

If you're on a Mac, you can toggle it on or off in your settings here - `System Settings` -> `Accessibility` -> `Display` -> `Reduce motion`

If you're on Windows, you can toggle it on or off in your settings here - `Settings` -> `Accessibility` -> `Visual effects` -> `Animation effects`

Take a look at Can I use... prefers-reduced-motion for information about which browsers support this feature

How to support it

It's like any other media query (such as screen size). The two options available are `reduce` or `no-preference`. You can use it in CSS to turn off an element's motion.

If you're providing moving media to developers, also provide a non-moving variant that they can use in the media query.

If you're a developer, there are two ways you can use it. Either you can reduce the motion if the user has set a preference to reduce motion, like this:

```
.moving-element {
    animation: bounce 500ms;
}
@media (prefers-reduced-motion: reduce) {
    .moving-element {
        animation: none;
    }
}
```

Or you can set the motion to happen if the user hasn't set a preference. Here's an example of using it like this:

```
@media (prefers-reduced-motion: no-preference) {
    .moving-element {
        animation: bounce 500ms;
    }
}
```

This is the recommended way, because browsers that don't support the prefers-reduced-motion media query will ignore this code, so your element will have no motion as standard. It's also less code. People who haven't turned on the setting will still see the motion, in the same way as the first method.

Useful links

- Web Content Accessibility Guidelines (WCAG) 2.2 Techniques - Using the CSS prefers-reduced-motion query to prevent motion
- Respecting users' motion preferences
- MDN - prefers-reduced-motion

Presentations

What is it for?

Most people will have to give a presentation at some point. Presentations are used to convey information, so it's incredibly important that everyone can get the information being provided.

How to support it

There are several things you can do to make presentations more accessible. Many of these are covered in more detail in other highlights.

Here are a few suggestions:

- Attach your slides to the meeting invite, so everyone can see the slides in advance. This means that anyone who uses a screen reader can get the information, and anyone who learns better by reading than by listening can take the time to read the information.
- Include alt text on all images in your slides. In PowerPoint, right-click the image, choose 'View Alt Text', then type your alt text. In Google Slides, click the image, click the paintbrush icon that appears (if you hover over it, it'll say 'Format options'), then add alt text in the 'Alt text' section.
- Consider your font. Make sure it's at least 18pt in size, and use a sans serif font (Atkinson Hyperlegible Next is a good example).
- Use left-aligned text, as it's easier to read for the majority of people.
- Don't use images of text. If you can't avoid it (like a company logo), make sure you add alt text to it.
- Check your colour contrast. This is especially important when you're using brand colours as backgrounds. It's always worth checking your link colour if you've included any links, too.
- If you're including links, make sure the link text is descriptive - no 'Read more' links, please.
- Ensure that colour isn't your only way of conveying information. 1 in 12 men and 1 in 200 women are colour blind, so make sure they can see all the information you've put together.

Useful links

- Making presentations more accessible
- NHS England: Making meetings accessible

QR Codes

What is it for?

A QR (Quick Response) code is a type of two-dimensional barcode. It's a machine-scannable image that a smartphone camera can read to bridge the gap between a physical item and a digital resource.

QR codes are used to store information for a device to read instantly. Common uses include:

- Opening a website
- Connecting to a Wi-Fi network
- Making a payment
- Downloading an app

While QR codes can help users with motor or cognitive disabilities by removing the need to type long uniform resource locators (URLs), they create significant barriers if not implemented carefully. These barriers include:

- Users who are blind or have low vision cannot easily locate or scan a QR code
- Poor placement can make codes unreachable for wheelchair users
- A standard code provides no information about its destination before being scanned
- Low-contrast colours or small sizes make codes difficult to see and scan

How to support it

To ensure your QR code is accessible, you must provide alternatives and context.

- Always include a short, readable URL (e.g., AbiHarrisonNye.dev/talks) and a clear description of the link's purpose next to the code
- Use high-contrast colours (dark on light), a large size (at least 1 inch x 1 inch), and place it at a height reachable by all users

If you can avoid using a QR code, you should. Here are a few alternatives:

- Short URLs - A simple, memorable link is often the most inclusive option
- NFC Tags - These allow users to tap their phone instead of aiming a camera, which is far easier

- Accessible QR (AQR) codes - A new technology that uses special patterns to help apps guide visually impaired users to the code

Useful links

- Accessibility and QR codes
- What is an Accessible QR (AQR) code?
- Making QR codes more accessible
- Are QR Codes Accessible for People with Disabilities?
- Accessible QR Codes – The Ultimate Guide

Screen reader only styling

What is it for?

Screen reader only styling ensures that everyone who uses a website can access the important information it contains. In most cases, this is achieved with semantics, alt text and aria labels. However, in some cases, something should only be 'seen' by screen readers.

For instance, a blog page might have several 'Read more' links. Sighted users can see which article each link is for, but a screen reader user moving from link to link would only hear 'Read more, Read more, Read more'. You can add extra words to each link, such as 'about accessible meetings', that are hidden visually but still read out by screen readers.

How to support it

You can add the following to your CSS, LESS, SASS, or any other CSS-based styling:

```
.sr-only {
   border: 0;
   clip: rect(0 0 0 0);
   height: 0.0625rem;
   margin: -0.0625rem;
   overflow: hidden;
   padding: 0;
   position: absolute;
   width: 0.0625rem;
   white-space: nowrap;
}

```

When you're using things like display: none; consider whether a screen reader user would benefit from the thing you're hiding. That's a great time to use this styling!

Useful links

- WebAIM - Invisible content
- CSS Tricks - Inclusively Hidden

Screen Readers

What is it for?

Screen readers are pieces of software that announce content audibly. This allows a computer to read text, images, icons, and much more. Some screen readers are available for free, either pre-installed as part of your operating system or available for download from websites. Screen readers are used if someone does not have enough vision to be able to view content from a screen.

Screen readers are impactful to everyone, but especially for the following people:

- People with visual impairments
- People with learning difficulties
- People with cognitive disabilities
- Non-native speakers
- People who prefer to listen to content instead of reading it visually

How to support it

Most devices come with a screen reader installed as standard, so you can test your own pages: VoiceOver on Mac and iPhone, Narrator on Windows, and TalkBack on most Android devices.

When testing, check that:

- Decorative images and icons aren't read out
- All images are understandable using the alt text, without needing to look at the page
- The order of the page makes sense
- Everything on the page is fully functional using a screen reader
- There is a skip link that lets keyboard users, including screen reader users, skip the header navigation

Useful links

- An Introduction to Screen Readers
- RNIB: Screen reading software

Shifting Left

What is it for?

Imagine baking a cake, pulling it out of the oven, and realising you forgot the sugar. Trying to shove sugar into a fully baked cake is messy, stressful, and ruins the cake! It's much easier to just add the sugar at the beginning.

In technology, the timeline goes from left to right (Plan ➔ Design ➔ Build ➔ Production). Shifting left means moving accessibility checks to the very beginning of the project (the left side), instead of waiting until the end to test it. Finding an accessibility bug during the design phase is incredibly cheap and fast to fix. Finding it right before the website goes live is expensive and frustrating!

Shifting left is a team habit, built by teams talking to each other and asking questions early. No special software is needed.

How to support it

Everyone on the team plays a part right from the start! Here are some examples of how different roles can shift left:

- Designers can check colour contrast and plan out where keyboard focus will go before sending pictures to the developers.
- Developers can plan out which Semantic HTML tags to use before writing any complex code.
- Product Managers can make sure accessibility is listed as a requirement for the feature to be considered Done.

Shifting left can sometimes feel like it slows a project down at the very beginning because you are doing more planning and having more meetings. However, this early planning saves a massive amount of time and money at the end, because you never have to tear down and rebuild a 'baked cake'!

Useful links

- Deque: Shift Left Accessibility Testing
- Shift-left accessibility: Embedding inclusion early in agile projects

Social Media: Alt Text

What is it for?

On social media, images often hold the core message or punchline. Alt text ensures that context isn't lost for anyone by serving a few vital purposes:

- Allows screen readers to describe the visual content to visually impaired users
- Displays text descriptions when poor signal or low data prevents images from loading
- Helps users with cognitive disabilities better understand complex graphics or memes
- Boosts discoverability by helping platform algorithms understand and index your posts

How to support it

On LinkedIn, add the image and click ‘ALT’ at the bottom right.

On Facebook, add the image, click the three dots in the top right, and select ‘Edit alt text’.

On X (Twitter), add the image and click the ‘ALT’ box at the bottom.

In workplace chat tools like Slack, upload the image, click ‘Edit file details’ in the top left, and enter your text in the description box.

Useful links

- Alternative text on social media
- Alternative text in action
- Social Media and Alternative Text: A Reflection

Social Media: Custom Fonts

What is it for?

People often use generator websites to make their text look unique (e.g. 𝒸𝓊𝓇𝓈𝒾𝓋𝑒 or 𝐛𝐨𝐥𝐝). However, these are not real fonts. They are mathematical symbols from the Unicode standard designed for equations, not language.

When a screen reader encounters this text, it does not read the word. It reads the mathematical description of every single character. 𝐻𝑒𝓁𝓁𝑜 ("Hello") becomes "Mathematical script capital H, mathematical script small e..." which renders the message completely unintelligible.

How to support it

The kindest thing you can do is avoid these generators entirely for your display name, bio, and captions.

- Use the default keyboard font provided by the platform. This ensures your text is readable and searchable.
- Since you cannot use bold or italics on most apps, highlight key words by capitalising only the first letter (for example, 'this is Really important') instead. Avoid writing whole words in capital letters, as they're harder to read.

Useful links

- Accessible Social - Just Not My Type

Social Media: Emojis

What is it for?

Emojis add personality and tone to messages, but for a screen reader user, they are read aloud as literal text descriptions. For example, a simple smiley face is read out as 'Smiling face with open mouth'.

When emojis are used excessively or incorrectly, this feature can become overwhelming. If you string ten 'Clapping hands' emojis together, the user has to listen to the phrase 'Clapping hands' repeated ten times in a row. Similarly, if you use an emoji as a bullet point, the user hears the emoji name before every single item on the list, which distracts from the actual content.

How to support it

To keep your content fun but readable, follow these best practices:

- Use them sparingly. One or two emojis are usually enough to convey a mood.
- Put them at the end. Place emojis at the end of a sentence or paragraph so they do not interrupt the flow of reading.
- Never replace a word with an emoji (e.g. writing 'I want [pizza emoji]' instead of 'I want pizza'). This ensures the sentence still makes sense if the emoji description is different from what you expect.
- Check the emoji description. Use a site like Emojipedia to check exactly how your chosen emoji will be described to a blind user.

Useful links

- Emojipedia
- Accessible Social - Emoji
- The Good, The Bad, And The Ugly: Emojis and Accessibility

Social Media: Hashtags

What is it for?

Imagine trying to read a sentence where all the spaces have been removed. It would look like a jumble of letters that takes real effort to decipher. That is exactly what a screen reader encounters when it reads a hashtag written entirely in lowercase.

If you write #superbowl, a screen reader might read it as "Superb Owl" rather than "Super Bowl". By capitalising the first letter of each word, we give the technology a clear signal of where one word ends and the next begins. This ensures the hashtag is pronounced correctly and is easier for everyone to read visually.

How to support it

This technique is often called CamelCase or PascalCase. The rule is simple: always capitalise the first letter of every word within the hashtag.

- Do: #SocialMediaMarketing
- Don’t: #socialmediamarketing

It's a small habit change that makes a massive difference to clarity.

However, while capitalisation solves the pronunciation issue, it does not solve the issue of length. A hashtag that consists of ten different words is still difficult to process, even with capital letters.

Additionally, placing hashtags in the middle of a sentence can disrupt the reading flow for assistive technology users. It is generally best practice to group your hashtags at the very end of your post or caption.

Useful links

- Accessible Social
- RNIB - How to make your social media accessible
- Hootsuite - Inclusive Design for Social Media

Tap Targets

What is it for?

You should ensure that interactive elements like buttons or links are large enough and have enough space around them, to make them easy to press without accidentally overlapping onto other elements.

Small tap targets can cause rage clicks, where the user has to try to click on an interactive element more than once. Larger tap targets can remove barriers for a number of users, including (but not limited to):

- Users who use a mobile device where a touchscreen is the primary mode of interaction
- Users with mobility impairments, such as hand tremors
- Users who use a mobile device in environments where they are exposed to shaking, such as public transportation
- Users who find fine motor movements difficult
- Users who access a device using one hand
- Users with large fingers, or who are operating the device with only a part of their finger or knuckle
- Users who have low vision may better see the target

You don't need any special code for this. Just make sure your tap targets are large enough, with enough of a gap between them.

How to support it

Web Content Accessibility Guidelines (WCAG) 2.2 requires tap targets to be at least 24px x 24px at Level AA (SC 2.5.8 Target Size (Minimum)), unless there's enough space around them. At Level AAA, they must be at least 44px x 44px (SC 2.5.5 Target Size (Enhanced)). This is so that a user's finger pressing on one tap target does not inadvertently touch another tap target.

Note: `px` *means* `pixels`

Google's Material Design recommends a minimum tap target size of 48px x 48px, with a gap of at least 8px between tap targets. This is larger than the minimum required by WCAG.

The 48px x 48px area corresponds to around 9mm, which is about the size of a person's finger pad area.

Any interactive element (unless it's text links within a paragraph) should be designed and built to be a minimum size of 48px x 48px, with a gap of 8px between any tap targets.

```
button {
    min-width: 48px;
    min-height: 48px;
    margin: 8px;
}
```

While an icon contained by a link may only have a width and height of 24px, you can add padding to the link to bring the tap target size up to 48px.

```
a {
    display: inline-block;
    margin: 8px;
    padding: 12px;
}
```

Useful links

- Understanding SC 2.5.8: Target Size (Minimum)
- How to use typography, touch targets, and spacing for better accessibility
- Accessible tap targets cheat sheet

Temporary & Situational Disabilities

What is it for?

People often link accessibility with permanent disabilities. For example, someone born without the use of their hands, or someone who lost their sight later in life. But disability exists on a much wider spectrum.

A temporary disability will go away over time, like a broken arm or an eye infection that makes screens painful to look at.

A situational disability is when your surroundings create the same barriers as a disability, even though nothing is wrong with your body. Trying to read a phone in bright sunlight makes even a clear, high-contrast screen almost impossible to see. Being in a loud pub makes it nearly impossible to follow any audio content without captions.

Designing for accessibility doesn't just help disabled people, who make up around 1 in 4 people in the UK. It improves the experience for everyone because at some point in their lives, most people will find themselves in one of these situations.

Temporary and situational disabilities are a key part of the Microsoft Inclusive Design framework. It describes permanent, temporary, and situational impairments to movement, vision, hearing, and speech. Many Web Content Accessibility Guidelines (WCAG) rules, such as captions, good colour contrast, and touch target sizes, also help people in these situations. This is true even when the rules weren't written with them in mind.

How to support it

Thinking beyond permanent disability unlocks a much bigger picture. Designing for the extremes means everyone in the middle benefits too.

- Designers can ask 'what if someone is doing this in harsh sunlight?' when checking colour contrast. This helps someone with low vision just as much as someone on a sunny day.
- Developers can add captions to all video content. Essential for someone who is d/Deaf, but also useful for anyone watching without headphones.

- Product Managers can include situational scenarios in user stories, like 'a user completing this task while commuting', so real-world constraints are considered from the start.

The more teams think this way, the harder it becomes to see accessibility as a niche concern. Designing for someone holding a baby also means designing for someone with a broken arm, and for someone with a permanent upper limb disability. One solution, three problems solved.

Useful links

- Microsoft Inclusive Design: Recognizing Exclusion
- World Wide Web Consortium (W3C): Diverse Abilities and Barriers

Text Alternatives for Complex Images

What is it for?

Standard 'alt text' is great for simple photos, but it often fails for complex data like charts, graphs, flowcharts, or infographics. A single sentence cannot convey all the information provided. People using screen readers (or anyone who struggles to process dense visual data) must be able to access the same information as everyone else.

How to support it

It requires a two-step approach because a short description is not enough

1. Use the standard 'alt text' to briefly identify the image (e.g. 'Bar chart showing Q3 revenue')
2. Provide the full data in a text format nearby

For charts and graphs, place a properly formatted data table immediately below the image.
For infographics, provide a full text transcript or a link to a page that outlines all the information in the same order.
For diagrams, write a structured list or description that walks the user through the flow.

Useful links

- WAI - Complex Images
- WebAIM - Accessible Images

Time Limits

What is it for?

Time limits refer to when a page sets a time limit on doing a certain action. This is often seen while filling in forms, or whilst making a purchase. When using time limits, allow sufficient time for users to complete tasks, read content, or interact with digital interfaces.

To make time limits accessible, Web Content Accessibility Guidelines (WCAG) 2.2 has a Level A requirement (SC 2.2.1 Timing Adjustable). Users must be able to do at least one of these things: turn the time limit off, change it to at least 10 times its length, or get a warning before it runs out and extend it with a simple action, such as pressing a button. They must have at least 20 seconds to respond to the warning, and be able to extend the time at least 10 times. There are a few exceptions, such as live auctions.

People with cognitive disabilities, and people who use screen readers, often need more time to take in information and finish tasks. A fixed time limit can cause pressure and anxiety. This can break people's concentration and stop them finishing, whether or not they have a disability.

Time limit considerations are most important for:

- Users with motor skills impairments
- Users with cognitive difficulties
- Screen reader users

How to support it

Ensure that any time limits you use can be manually extended. Also ensure that the implementation of those time limits isn't done in a scary or anxiety-inducing manner.

Useful links

- Access Guide: Time limits
- Understanding SC 2.2.1: Timing Adjustable
- Web Accessibility Tips: Give People Enough Time

Tooltips & Popovers

What is it for?

Tooltips and popovers give users extra info on demand (like for an icon button). But they often fail accessibility by being:

- Mouse-only. They don't appear for keyboard users (focus).
- Silent. Screen readers don't announce the new content.
- Hard to close. Users get 'stuck', or the tooltip vanishes too fast.

It's important to note that popovers and modals are not the same thing. A modal demands your full attention, while a popover just offers extra information.

How to support it

They must work for keyboard, mouse, and screen reader users.

- For simple tooltips (read-only info):
    - Show on hover and focus
    - Keep it visible while the pointer is over it, and let users close it with the Escape key
    - Use aria-describedby to link the trigger button to the tooltip. This is how screen readers announce it
- For complex popovers (with buttons/links inside):
    - Trigger on click (not hover). This is more robust
    - Manage keyboard focus: Move focus into the popover when it opens, and return it to the trigger button when it closes. Don't trap focus inside, as that's only for modals
    - Make it easy to close: Support the Escape key and include a visible close button

Useful links

- Accessible Rich Internet Applications (ARIA) Authoring Practices - Tooltip Pattern

Touch Gestures

What is it for?

Imagine trying to open a heavy door, but the rule is you can only use your pinky fingers. For someone with shaky hands, or someone who uses a special typing wand held in their mouth, doing a pinch-to-zoom or a three-finger swipe on a screen is just as hard. If a map app only lets you zoom in by pinching, those users are completely stuck.

How to support it

Always give users a simple one-finger tap way to do things.

- If there's a map you can pinch to zoom, also add simple buttons that can be tapped (like + and -)
- If there's a side panel you have to swipe to open, also add a visible Menu button to tap
- Make sure no important action requires twisting, pinching, or using multiple fingers at the same time

You don't have to get rid of fun swipes and pinches completely. Those complex gestures are fantastic shortcuts for people who can easily perform them. However, make sure they're not the only way to get the job done. Every fancy shortcut needs a standard, simple path too.

Useful links

- World Wide Web Consortium (W3C) - Understanding Pointer Gestures
- WebAIM - Motor Disabilities

Zooming

What is it for?

Some people need to make the text and images on a website bigger to see them clearly. Pages must support this. When a user zooms in, the content should automatically rearrange itself to fit the screen. This is often called 'reflow'. A good page will reflow so that users never have to scroll from side to side to read a single sentence. This is very important for creating a positive experience for people with low vision.

All modern browsers support zooming. It's up to developers and designers to build layouts that still work when people zoom in.

How to support it

You can easily test if your page is zoomable, using the following steps:

On a computer:

1. Hold the Ctrl key (or Cmd on a Mac) and press the + key.
2. Zoom to 200% (twice the normal size) and check everything is readable and usable.
3. Then zoom to 400%. Can you read everything without scrolling left and right? All information and buttons should still be visible and easy to use. Do all navigation menus and search bars work correctly?

On a mobile (known as 'pinch-to-zoom'):

1. Put two fingers on your screen, close to each other
2. Spread your fingers apart, keeping them touching the screen
3. Release at your desired zoom level

NOTE: This is especially important on mobile devices. The 'pinch-to-zoom' feature must never be blocked. Disabling zoom creates a major barrier for accessibility and prevents people from using a tool they may rely on.

Useful links

- Adrian Roselli - Don't Disable Zoom

- CSS-Tricks - Accessible Font Sizing, Explained
- WebAIM - Responsive Design and Reflow

Glossary of abbreviations

This glossary gives the full form of the abbreviations used in this document.

- **a.k.a.**: Also known as
- **A11y**: Accessibility (a numeronym: 11 letters between the 'a' and the 'y')
- **AA and AAA**: WCAG conformance levels. AA is the middle level; AAA is the highest
- **ADHD**: Attention deficit hyperactivity disorder
- **Alt text (ALT)**: Alternative text
- **AMOLED**: Active-matrix organic light-emitting diode (a type of screen)
- **AQR**: Accessible QR code
- **ARIA**: Accessible Rich Internet Applications
- **ASCII**: American Standard Code for Information Interchange
- **AT**: Assistive technology
- **CSS**: Cascading Style Sheets
- **e.g.**: For example
- **etc.**: And so on
- **HMRC**: HM Revenue and Customs
- **HTML**: HyperText Markup Language
- **i.e.**: That is
- **LESS**: Leaner Style Sheets (a CSS preprocessor)
- **MDN**: MDN Web Docs (originally the Mozilla Developer Network)
- **NFC**: Near-field communication
- **OLED**: Organic light-emitting diode (a type of screen)
- **PAC**: PDF Accessibility Checker
- **PDF**: Portable Document Format
- **pt: Points (a unit of font size)**
- **px**: Pixels
- **Q3**: The third quarter of the year
- **QR**: Quick Response (as in QR code)
- **RNIB**: Royal National Institute of Blind People
- **SASS**: Syntactically Awesome Style Sheets (a CSS preprocessor)
- **SC**: Success criterion (a single testable WCAG requirement)
- **SEO**: Search engine optimisation
- **TV**: Television
- **UK: United Kingdom**
- **URL**: Uniform Resource Locator (a web address)

- **W3C**: World Wide Web Consortium
- **WAI**: Web Accessibility Initiative
- **WCAG**: Web Content Accessibility Guidelines

Glossary of terms

This glossary explains technical terms used in this document.

- **Ableist language**: Words that use disability as an insult or a negative image, such as 'lame' or 'blind spot'
- **Accessibility debt**: Accessibility problems that build up when accessibility is left until later
- **Accessibility overlay**: A third-party add-on that claims to fix a website's accessibility problems automatically
- **Alt text**: A short written description of an image, read out by screen readers
- **Assistive technology**: Tools that help disabled people use a device, such as screen readers, magnifiers, switches and voice control
- **Autocomplete / autofill**: When a browser or password manager fills in saved details for you
- **Biometrics**: Using a fingerprint or face scan to sign in
- **Breadcrumbs**: A trail of links showing where a page sits on a website, such as Home > Phones > iPhone
- **CamelCase**: Writing each word in a hashtag with a capital letter, such as #SuperBowl
- **Captions**: Text on screen showing the speech and important sounds in a video
- **Carousel**: A banner that moves through a set of images or messages
- **Cognitive**: To do with thinking, memory, attention and understanding
- **Colour contrast ratio**: How different two colours are in brightness. A higher number is easier to read, such as 4.5:1
- **Component library / design system**: A shared set of ready-made, tested parts, such as buttons and forms, used to build products
- **Decorative image**: An image that adds nothing to the meaning, so screen readers can skip it
- **Focus**: Which element is currently active and will receive keyboard input
- **Focus state**: How an element looks when it has focus, such as a visible outline
- **Focus trap**: When keyboard users can't tab away from part of a page
- **Gesture**: A touch action, such as a swipe, pinch or multi-finger tap
- **Hover state**: How an element looks when a mouse pointer is over it
- **Identity-first language**: Putting the identity first, such as 'an autistic person'
- **Interactive element**: Anything you can click, tap or type into, such as a link, button or form field

- **Landmark**: A labelled area of a web page, such as the navigation or main content, that screen reader users can jump to
- **Luminance**: How bright a colour looks
- **Magic link**: A one-time sign-in link sent by email
- **Media query**: A rule in CSS that changes how a page looks based on the device or the user's settings
- **Modal**: A pop-up window that appears on top of a page and must be closed before you go back to the page
- **Motor impairment**: A disability that affects movement, such as tremors or limited use of the hands
- **Neurodiversity**: The natural differences in how people's brains work, including autism, ADHD, dyslexia and dyspraxia
- **One-time code**: A short code sent by text or app that you use once to sign in
- **Person-first language**: Putting the person first, such as 'a person with vision loss'
- **Persona**: A made-up but realistic person used to test whether a product works for different people
- **Photosensitive epilepsy**: Epilepsy where seizures can be triggered by flashing or flickering light
- **Reflow**: When a page rearranges itself to fit the screen, so you don't need to scroll sideways
- **Screen magnifier**: Software that enlarges part of the screen
- **Screen reader**: Software that reads out what is on the screen
- **Semantic HTML**: Using the HTML element that matches the content's meaning, such as <button> for a button
- **Shifting left**: Thinking about accessibility at the start of a project rather than the end
- **Skip link**: A link at the top of a page that lets keyboard and screen reader users jump past the navigation
- **Switch device**: A single button or similar device used to control a computer, for people with limited movement
- **Tagged PDF**: A PDF with hidden structure (headings, lists, reading order) that screen readers can use
- **Tap target**: The area of a button or link you can tap or click
- **Tooltip / popover**: A small box of extra information that appears next to an element
- **Transcript**: A written version of everything said, and important sounds, in audio or video

- **Vestibular disorder**: An inner-ear condition that affects balance, which motion on screen can make worse
- **Voice control**: Controlling a device by speaking commands