2.0 AA / 2.1 AA / 2.2 AA

WCAG Guide

Kris Rivenburgh

Need to learn *faster*?

My WCAG Course has code examples, videos, and workbooks.

- Covers version 2.0 AA, 2.1 AA, and, 2.2 AA.
- Save 20+ hours in implementation
- Catered to individuals and teams

One license gets you access forever, no subscription.

Need to learn even faster? Hire me for consulting.

Copyright

Attribution

1) **This checklist is not affiliated with the W3C.** When written, this checklist, myself, my company (Kris's WCAG 2 Guide, Kris Rivenburgh, and Accessible.org, LLC) are not affiliated or associated with W3C, WAI, WCAG, or any W3C initiatives in any way.

2) **WCAG is the Source of Material**. The W3C and WAI gets all credit for any material ideas, recommendations, guidelines, specifications, etc. in this checklist. All I have done is condensed, rewritten, interpreted, and presented WCAG's material information in an informal, less technical manner.

Here is full attribution to W3C:

Accessibility

PDF/UA and WCAG 2.0 AA standards were followed in the creation of this document.

If you have any difficulty accessing the contents of this document, contact me at kris@accessible.org and I will ensure that you have full access to this checklist.

Disclaimer

By reading this document, you agree to release Author/Creator/Publisher, Kris Rivenburgh and Accessible.org, from any and all liability resulting from your use of this document.

This checklist is an interpretation of WCAG 2.1 AA success criteria by one person. This will not fully convey all information found in WCAG 2.1 AA. Moreover, this checklist may be inaccurate or misrepresentative of the information contained in WCAG 2.1 AA.

By continuing to read this checklist, you understand that it will be incomplete and may be inaccurate. You agree not to hold the Author/Creator/Publisher of this checklist (or any of his businesses/companies) liable for any reliance upon this document.

You further agree to base your ultimate web accessibility decision-making on the original WCAG 2.1 AA source document itself and not this checklist.

The material in this checklist is provided "as is" and without warranties of any kind express or implied. The Author/Creator/Publisher disclaims all warranties, express or implied, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose. The Author/Creator/Publisher does not warrant or make any representations regarding the use or the results of the use of the materials in this checklist in terms of their correctness, accuracy, reliability, or otherwise.

Under no circumstances, including but not limited to, negligence, shall the Author/Creator/Publisher be liable for any special or consequential damages that result from the use of, or the inability to use this checklist. In no event shall the Author/Creator/Publisher's total liability to you for all damages, losses, and causes of action exceed the amount paid by you, if any, for this checklist. You agree to hold the Author/Creator/Publisher and any connected principals, agents, or entities free from any and all liability for all claims for all damages except for claims of gross negligence and intentional wrongdoing. Note that applicable law may preclude the exclusion of implied warranties or the limitation of damages so these may not apply to you.

You agree that any and all claims for gross negligence or intentional tort shall be settled solely by confidential binding arbitration per the American Arbitration Association's commercial arbitration rules. All arbitration must occur in the municipality where the Author/Creator/Publisher's principal place of business is located, Dallas, Texas at the time of publication. Your claim cannot be aggregated with third party claims. Arbitration fees and costs shall be split equally, and you are solely responsible for your own attorney's fees.

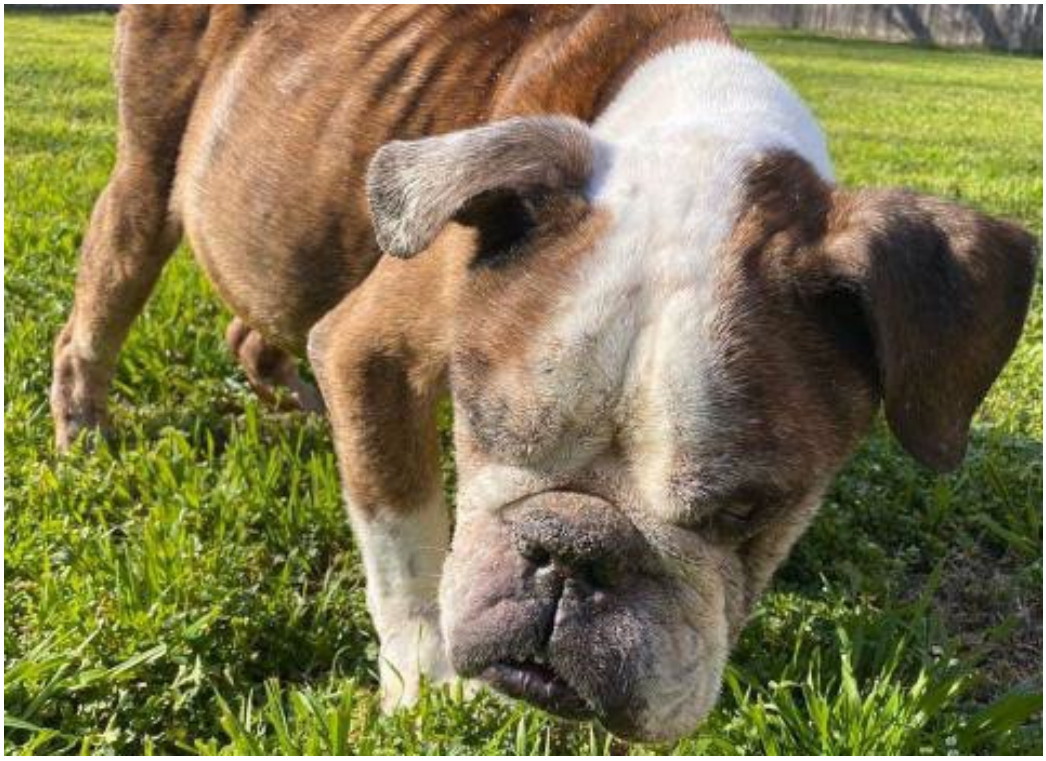
Your use of this checklist and engagement of any activities mentioned in this checklist means that you are legally bound to these terms. You should always conduct your own due diligence engaging in activity that potentially has large financial risks or expenditures.

No links to external third-parties constitutes an endorsement or recommendation of any kind. Keep in mind that domains expire and pages are taken down so some links may no longer work (or may go to unknown websites) when you access this checklist.

If you have any questions or reservations about the legal terms, conditions, and/or copyright of this checklist, contact me at kris@accessible.org.

As a condition to use of this document, you must agree to all terms herein prior to use of this document. By continuing, you agree to all legal terms, disclaimers, copyright, and other conditions.

Table of Contents



In Memory of Groucho

Groucho had been abandoned by his previous owners and we took him on as a hospice foster, hoping to give him the best life possible in whatever time he had left. Groucho only made it 36 hours but in his brief time he became part of the family and was a great dog.

My favorite memory of Groucho was sitting with him on the back porch while I read and he soaked up the sun beside me.

I love that Groucho will now be known by people all over the world. And I love that a big part of his legacy will be helping fellow dogs get adopted.

There are so many amazing senior dogs at your local shelter or rescue. Look for The Alma and Archer Foundation in 2023. This will be a different kind of dog rescue.

What is WCAG?

Web Content Accessibility Guidelines (WCAG) is a set of technical standards to ensure that digital assets (e.g., websites, apps) are accessible to persons with disabilities.

WCAG is authored by the **Web Accessibility Initiative (WAI)** under the W3C.

- **Version 2.0 (2008)**. This classic version is a strong baseline in accessibility.
- **Version 2.1 (2018)**. This **current version** includes key mobile criteria.
- **Conformance Levels**: A, AA, AAA

Level AA is required for legal compliance across the world.

To be fully WCAG conformant, your digital asset must meet all success criteria under a given version and conformance level.

- **2.0 AA**. 38 success criteria.
- **2.1 AA**. 12 new criteria.
- **2.2 AA**. 7 new criteria.

Each level is cumulative: **2.1 AA** conformance will satisfy **2.0 AA**.

Important: About This Guide

This guide provides a succinct explanation for the 55 AA criteria across all WCAG versions.

Think of this guide as a really good cheatsheet – it doesn't cover every single detail, but you can still get a B+ on the exam.

To fully understand each criterion:

- Read the W3C source documentation
- Watch my WCAG course

Key Concepts

Terms

This guide is for anyone — developers, marketers, designers, content managers.

But it helps if you understand these terms:

- **HTML** - Creates and structures web content. It defines elements, such as headings, paragraphs, links, and images.
- **CSS** - Controls the styling of web pages in HTML. It defines how elements should be displayed (e.g., layout, colors, fonts, spacing).
- **ARIA** - Attributes that provide extra details about elements that may not be conveyed by HTML alone (e.g., name, role, state, value).
- **Screen Readers** - Technologies that read aloud web pages or documents to users who are visually impaired or blind.

Avoid Overlays or Widgets

Some applications open a menu of accessibility options on your website.

They claim to instantly fix accessibility, but this claim is false. Full WCAG conformance can only be achieved with manual remediation.

By continuing to read this guide, you agree that overlays do not work and promise not to use them.

For more information, see OverlayFactSheet.com and OverlaysDontWork.com.

Criteria

1.1.1 / WCAG 2.0 / Level A

Non-Text Content

Provide text alternatives for all non-text content (e.g., images, maps, art, photos, infographics, graphs).

Steps

1. Add alt text and/or descriptions to meaningful images and ensure they convey the same information as the visual content.
2. Add empty alt attributes for decorative images (i.e., non-meaningful images).

Tips

- **Concise and Descriptive**. Describe what the image means or represents in as few of words as possible; what is the key takeaway?
- **Length**: If you need more than 125 characters, consider using aria-describedby or longdesc for a fuller description.
- **Context**. Context counts. The same image may need different alt text based on its purpose on each page.
- **Redundancy**. Avoid descriptions that repeat other text on the page.
- **Complex Images**. For complicated images (e.g., infographics), provide a general description and reference the full text description elsewhere.

Examples

- **Product Images:** “stainless steel coffee maker with removable spot and liquid catch underneath”
- **Graphic Images:** “chart showing 10% quarterly sales growth from 2023”
- **Functional Images:** search icon with alt text “search”
- **Decorative Images**: empty alt="" for background patterns or purely aesthetic elements
- **Complex Images**: "Q4 sales infographic, detailed breakdown in following section"

Exceptions

- Audio and video are covered in other criteria.
- Tests or exercises that would be negated if presented in text (e.g., a hearing or visual accuracy test).

1.2.1 / WCAG 2.0 / Level A

Audio-Only and Video-Only (Prerecorded)

Provide text transcripts for audio-only content or text transcripts or audio descriptions for video-only content.

Steps

1. Provide a full text transcript near audio or video files.
2. If you can't paste a full text transcript nearby, add a descriptive link that sends people to a full transcript on another page.

Tips

- **Audio Content**: Include speaker names, dialogue, and relevant sounds (e.g., door slamming).
- **Video Content**: Describe key visual information including actions, characters, scene changes, and text shown on screen.

Examples

- Podcasts (audio-only)
- Silent movies (video-only)
- Ambient soundtracks (audio-only)
- Animated GIFs (video-only)

1.2.2 / WCAG 2.0 / Level A

Captions (Prerecorded)

Provide captions showing all dialogue and important sounds for videos with audio.

Steps

1. Add synchronized closed captioning to all videos with audio.

Tips

- **Closed Captions > Subtitles**. Subtitles convey only dialogue, whereas closed captions include important sounds (e.g., door slamming). Subtitles assume that audiences can hear sound but may not be able to understand the dialogue; closed captions assume that audiences can't hear the audio.
- **Automatic Subtitles Are Imperfect**. Most tools (e.g., YouTube) will require additional editing to fix mistakes or add background information.
- **Caption Quality**: Ensure captions are accurate, synchronized with speech, and easily readable.

Examples

- Speakers with labels, such as [John] or [Narrator].
- Sound effects, such as [laughter], [applause], or [door slams].
- Music cues, such as [uplifting music playing] or [ominous tone].

Exceptions

- Silent or text-based videos
- Background music-only

1.2.3 WCAG 2.0 / Level A

Audio Description or Media Alternative (Prerecorded)

Provide text or audio descriptions of important visual content that isn't conveyed through the main audio track.

Steps

1. Include synchronized audio descriptions for visual information not covered in the main audio.
2. Another option is to provide all information in the video in text, but audio descriptions are required in 1.2.5 so you will need them regardless.

Tips

- **Provide All Meaningful Visual Information**. Include any relevant context (e.g., actions, scenery, expressions, laughter).
- **Extended Audio**: You can create an alternate version of the video with extended gaps between dialogue to fit audio descriptions.

Examples

- Scene change: "The scene shifts to a busy office"
- Action without dialogue: "John shakes his head no"
- Text on screen: "A '20% Off' sign appears"

Exceptions

- Pure audio
- Live streams
- Unimportant visuals

1.2.4 / WCAG 2.0 / Level AA

Captions (Live)

Provide captions for important audio content during live broadcasts.

Steps

1. Include real-time closed captions with formal, live broadcasts.

Tips

- **Caption Quality**: Live captions should be accurate and synchronized, though they may have a slight delay.
- **Formal Events**: This applies to planned events like webinars and live streams, not impromptu meetings.
- **Use Professional Caption Services**. Existing options will be easier than handling it yourself.
- **Automatic Captions**: While auto-captions have significantly improved because of AI, they may not be reliable enough for live events.

Examples

- Webinars
- News broadcasts
- Townhall broadcasts

Exceptions

- Impromptu personal social media streams
- Informal Zoom meetings

1.2.5 / WCAG 2.0 / Level AA

Audio Description (Prerecorded)

Provide audio descriptions for all prerecorded videos.

Steps

1. Provide audio descriptions in your video.

Tips

- **How It Works**: Audio descriptions are narrations added during natural pauses to describe important visual information not covered in the main audio.
- **Required**. While 1.2.3 allowed a text alternative OR audio descriptions, this criterion requires audio descriptions.
- **Extended Audio**: You can create an alternate version of the video with extended gaps between dialogue to fit audio descriptions.

Examples

- **Describing a Scene:** “A woman hurriedly sifts through a crowded market.”
- **Highlighting On-Screen Text:** “The sign reads: ‘No entry after 10 PM.’”
- **Detailing Actions:** “The dog jumps over the fence and chases after the car.”

1.3.1 / WCAG 2.0 / Level A

Info and Relationships

Ensure information, relationships, and structure conveyed through visual or audible presentation are available programmatically or through text.

Steps

1. Use HTML5 semantic markup.
2. Use ARIA attributes for custom components.
3. Add text to accompany audible cues.

Tips

- **Visual to Code**: Any information conveyed through visual formatting (size, color, layout) must also be conveyed through code and/or text.
- **Audio to Code**: Any audible notifications (e.g., chimes, announcements, etc.) need text equivalents that screen readers can announce.
- **Why Markup?** A sighted user can determine that large text is a heading, but a visually impaired user doesn't have this visual cue. HTML markup can convey this information.
- **Use ARIA**. ARIA labels can add meaning to dynamic, custom, or advanced applications where HTML5 is insufficient.

Examples

- Headings: use <h1> to <h6> to structure content
- Content type: Define content types by using <article>, <video>, etc.
- Lists: use ordered <ol> and unordered <ul> lists
- Form associations: connect labels to inputs
- Data relationships: associate table headers (th) and data cells (td)

1.3.2 / WCAG 2.0 / Level A

Meaningful Sequence

When the order of content affects its meaning, the correct reading sequence must be preserved in the code.

Steps

1. Structure your website's content programmatically so that it appears in a logical reading order for screen reader users.
2. Ensure CSS positioning doesn't change the logical reading order.

Tips

- **Typical Reading Order.** Website visitors typically read across this sequence: header, main content, sidebar, footer.
- **General Reading Order.** Top to bottom, left to right.
- **More Than One Way.** There can be more than one correct order. Just choose one.
- **Test Without CSS**. Some browser extensions can disable CSS to let you view a website without styling (and how screen readers would interpret the order of the content).
- **Watch Positioning**: Using CSS position:absolute or float can disrupt the natural reading order.

Examples

- Navigation menus read in correct order
- Multi-column content flows logically
- Form labels precede input fields

Exceptions

- Where order of content is irrelevant
- Interactive graphs or maps

1.3.3 / WCAG 2.0 / Level A

Sensory Characteristics

Write clear instructions that don't rely solely on shape, size, visual location, orientation, or sound.

Steps

1. Avoid sensory-only instructions (e.g., click the rectangle button).
2. Add text references (e.g., click the rectangle button labeled “subscribe”).
3. Using sensory instructions is okay, just make sure instructions are also available programmatically or through text.

Tips

- **Include Multiple Senses**. Multiple senses reinforce information and help users who rely on different senses.
- **Color References**: When using color in instructions, include non-color identifiers (e.g., "click the red Submit button" instead of "click the red button").
- **Screen Reader Users**: Remember that terms like "above," "below," and "to the right" may not be meaningful to screen reader users.

Examples

- Instead of instruction, “click the red button,” use, “click the red button labeled ‘Submit.’”
- Rather than instructing, “respond when you hear the chime,” write, “respond when you hear the chime or when you receive a text prompt saying ‘Answer now.’”

1.3.4 / WCAG 2.1 / Level AA

Orientation

Content should be accessible at any orientation (e.g., landscape, portrait).

Steps

1. Rotate your phone and check whether your website looks and functions properly in both orientations (i.e., horizontal and vertical views).

Tips

- **Avoid Locking Orientations**. Unless it's absolutely necessary.
- **Test All Breakpoints**: Ensure content remains accessible at all screen sizes and orientations, not just mobile devices.
- **Why Does It Matter?** Some devices let users lock the orientation. Plus, some devices are mounted at fixed orientations (e.g., monitor on a wheelchair).

Examples

- An app shouldn't force users to rotate their device to a specific orientation.
- Games that can be played in either orientation (unless the game mechanics require a specific orientation).

Exceptions

- Content that requires a specific orientation (e.g., depositing a bank check in which you need a horizontal landscape).
- Specialized tools that are designed to function in only one orientation (e.g., medical device).

1.3.5 / WCAG 2.1 / Level AA

Identify Input Purpose

Form input fields should have their purpose identified in code to enable autofill and help assistive technologies".

Steps

1. Identify which form fields collect standard personal information.
2. Add appropriate autocomplete attributes to input fields so browsers and assistive technologies can understand their purpose.

Tips

- **Use Standard Values**: Use predefined autocomplete values from HTML specification to ensure broad compatibility. Most personal information fields (name, email, address) and payment fields (card number, expiration) have standard autocomplete values.
- **Use Visual Indicators**. Icons such as birthday cakes can help visually indicate that a form field is prompting the user to enter their birthdate.

Examples

- A contact form that can autofill with saved browser data because the input purposes are correctly identified.
- Input field using autocomplete="email" for email address.
- Input field using autocomplete="postal-code" for zip code.

Exceptions

- Custom input fields
- CAPTCHAs or other security concerns
- Unstructured or free-text input fields (e.g., message inputs)

1.3.5 / WCAG 2.0 / Level A

Use of Color

Color should not be the only way to convey information.

Steps

1. Identify all instances where color is used to convey information to the user (e.g., a pie chart that is color coded with its legend). Add text or another way to provide an alternative means of accessing information.

Tips

- **Add text or another way**. Make sure text (or symbols, textures, etc.) accompanies information relayed by color.
- **You Can Still Use Color**. Simply account for users who may not be able to distinguish colors.
- **Use Patterns and Textures**. Add diagonal lines or other patterns in bar charts and graphs to provide another means of ascertaining information without color.
- **Equivalent**. The alternative means provided should convey the same level of information as the color coding.

Examples:

- Required fields in red and asterisk (*)
- Error messages in red and warning marker (e.g., !)
- A progress bar that uses color and numerical percentages.
- Blue links with additional marker (e.g., underline)
- A map that uses color to distinguish regions with a text key for context.
- Toggle switches in green vs. gray with text labels (e.g., “on” and “off”)

1.4.2 / WCAG 2.0 / Level A

Audio Control

Users must have a way to pause or stop any audio that automatically plays for more than 3 seconds.

Steps

1. Avoid using audio or video that plays automatically.
2. If media plays automatically, let users adjust, mute, pause, or stop it.

Tips

- **Give Users Control**. Put users in control of content.
- **User Initiated**. Let users initiate audio after they reach a page rather than requiring them to stop audio.
- **Ease Of Use**. Audio controls should be easy to locate and operable with assistive technologies.

Examples

- Videos that play automatically but can be paused or stopped.
- Background music that starts automatically, but has an option to mute.
- Ads in an embedded media player with a mute or stop option.

Exceptions

- Audio that plays for less than 3 seconds.

1.4.3 / WCAG 2.0 / Level AA

Contrast (Minimum)

Text and images of text should stand out against their background by meeting a color contrast ratio of 4.5:1.

Steps

1. Regular text should have a 4.5:1 ratio.
2. Large text (14 point + bold or 18 point) should have a minimum 3:1 ratio.

Tips

- **Varying Backgrounds**. Remember to account for backgrounds that vary in color and ensure continual sufficient contrast.
- **Use Color Selector Tools**. Download a tool (e.g., Colorzilla) to grab the hexadecimal codes for colors (e.g., #efefef). Then enter this code into the WebAIM checker along with the background hex code to check the ratio.

Exceptions

- Logos or brand names
- Decorative text

Examples

- Text has a background overlay that rests over a gradient or image background to ensure sufficient contrast.
- In the image below, the white footer text on the bottom is easier to read because it has a higher contrast ratio.

1.4.4 / WCAG 2.0 / Level AA

Resize Text

All text can be resized up to 200% without any loss of content or functionality.

Steps

1. Increase the size of your text on your website by 200% and ensure that all content still displays properly.
2. There should be no difficulty in reading content or accessing functionality as a result of the increased text size.

Tips

- **Text Zoom vs. Page zoom**. Text zoom only increases the size of the text, not the entire page.
- **All Functionality**. Resizing text should preserve all functionality, such as the ability to use buttons, dropdown menus, and form fields, without overlap or inaccessibility.
- **Design Layouts with Relative Units**. Fixed pixels are less flexible than ems or percentages.
- **Use Min Height in CSS**. Containers should increase when text size increases.
- **Create a Default Starting Size**. A starting size of 14 pixels ensures that users don't need to zoom or resize text as much as they normally would.

Examples

- Avoid text overlapping on other text or elements and/or getting cut off

Exceptions

- Complex charts or diagrams
- Captions or subtitles in videos

1.4.5 / WCAG 2.0 / Level AA

Images of Text

Use actual text instead of images of text unless an exception is met.

Steps

1. Style text with CSS to preserve accessibility while maintaining visual design.
2. For instances where an image of text must be used, ensure the text is included as alternative text in the alt attribute. For complex images, consider providing a detailed text description elsewhere on the page.

Tips

- **Avoid Images of Text When Possible**. If you need to use an image of text, include an alt attribute with an alternative text description.

Exceptions

- Logos or decorative text
- Text appearance is critical to the message (e.g., a promotional poster with artistic typography). In these cases, ensure there is alternative text describing the content and purpose of the text.
- Complex images (e.g., graphs, screenshots): Provide text alternatives or accompanying descriptions to convey the same information.

1.4.10 / WCAG 2.1 / Level AA

Reflow

Users should be able to increase the size of content without needing to scroll horizontally.

Steps

1. View the page on a phone or set the desktop browser to 1280 pixels wide, then zoom to 400%. Does the website still function properly without scrolling horizontally?
2. Test content that includes interactive elements (e.g., forms, buttons, and navigation menus) to ensure they remain fully functional and do not overlap or become inaccessible when zoomed to 400%.

Tips

- **Build Responsive Pages**. It's easier to meet this criterion.
- **Make Sure Content Reflows**. Make sure content adapts to smaller screens.
- **Avoid Fixed-Width Containers**: Use flexible layout techniques such as percentages or CSS Grid to prevent horizontal scrolling.

Examples

- Text should wrap when the viewport is resized.
- Images should resize or reposition to fit the screen.
- Forms and input fields should remain fully visible and operable without horizontal scrolling.
- Navigation menus might stack vertically on smaller screens.

Exceptions

- Videos or multimedia with fixed aspect ratios that cannot adapt without distorting the content.
- Large or complex objects (e.g., maps, tables of data, etc.), where reflowing would reduce functionality or obscure meaning.

1.4.11 / WCAG 2.1 / Level AA

Non-Text Contrast

All meaningful non-text content, with exceptions, should have sufficient contrast with the color surrounding it.

Steps

1. Ensure that elements (e.g., buttons, forms, icons) have a minimum 3:1 color contrast ratio so they stand out.
2. Check all states of interactive elements (e.g., hover, focus, active) to ensure they maintain a 3:1 contrast ratio across all interactions.

Tips

- **Use A Color Picker**. You can extract the color hex code using a color selector and then paste the code into the WAVE color contrast checker.
- **Ensure Focus Indicators Are Visible**: Focus outlines or borders for interactive elements must also meet the 3:1 contrast ratio against adjacent colors.

Examples

- Buttons should stand out against the background.
- Social media icons should be easy to distinguish.
- Form fields should have a clear outline or background contrast to indicate where users need to input data.
- Toggle switches and sliders should have contrasting colors in both their "on" and "off" states.

Exceptions

- Logos
- Decorative elements
- Disabled controls

1.4.12 / WCAG 2.1 / Level AA

Text Spacing

Content should still appear and function properly if users change the text spacing (e.g., line height, letter spacing, word spacing).

Steps

1. Test your content by adjusting the CSS or using browser tools to modify the text spacing properties (line height, letter spacing, and word spacing). Make sure that content remains legible and accessible after these changes.
2. Line height should be at least 1.5x the font size.
3. Paragraph spacing should be at least 2x the font size.
4. Letter spacing should be at least 0.12x the font size.
5. Word spacing should be at least 0.16x the font size.
6. Ensure that all interactive elements (e.g., buttons, links, and form fields) remain accessible and do not overlap or lose functionality with increased text spacing.

Tips

- **Use Relative Font Sizes**. Specify font sizes with ems or rems, rather than fixed pixels.
- **Test With Tools**. Like Steve Faulkner's bookmarklet.
- **Use Flexible Containers**: Ensure containers adjust dynamically to accommodate increased spacing without cutting off content.

Examples

- Zoomed-in text remains visible inside containers without being cut off.
- Form labels and inputs remain aligned and functional when text spacing properties are increased.

Exceptions

- Complex data tables

1.4.13 / WCAG 2.1 / Level AA

Content on Hover or Focus

Users can keep or dismiss additional content that appears on hover or focus (e.g., tooltips, pop-ups, submenus).

Steps

1. **Dismissible**. Ensure users can close pop-ups or tooltips by pressing a key like "Esc" or moving the mouse away. This should work for both keyboard and mouse users.
2. **Hoverable**. Let users interact with content without it disappearing.
3. **Persistent**. Ensure that content remains visible until the user dismisses it, moves their pointer or keyboard focus away, or its visibility is no longer relevant.
4. Ensure the additional content is accessible via keyboard navigation and does not block access to underlying or adjacent content when displayed.

Tips

- **Add Sufficient Padding to Popup Elements**. Leave space on outside edges so that enlarged cursors can be placed here in order to read the new content without closing it.

Examples

- Tooltips that appear on hover, remain visible when hovered, and are dismissible via "Esc" or mouse.
- Dropdown menus that stay open on focus or hover and remain persistent through navigation.

Exceptions

- Content that is essential and cannot be hidden (e.g., error messages)

2.1.1 / WCAG 2.0 / Level A

Keyboard

All functionality should be focusable and operable through keyboard commands.

Steps

- Press the Tab key to move across elements on a web page (and Shift + Tab to move backward).
- Ensure all interactive elements can be activated using Enter or Space key.
- Check that any custom integrations can be fully operated with keyboard commands.

Tips

- **Disable the Mouse**. You should still be able to use all interactive elements.

Examples

- Navigating through a webpage using the Tab key
- Activating buttons or links using the Enter or Space key
- Accessing dropdown menus or sliders using arrow keys

Exceptions

- Interactive maps
- Drawing or design tools where keyboard input is impractical

2.1.2 / WCAG 2.0 / Level A

No Keyboard Trap

Users must be able to use a keyboard to navigate to and from all parts of a website.

Steps

1. Ensure users can navigate to all focusable elements using only a keyboard.
2. Ensure users can navigate both forward (Tab) and backward (Shift+Tab) without becoming stuck in any component or area, such as modals or widgets.

Tips

- **Check Integrations**. Keyboard accessibility issues are often introduced with embedded third-party integrations (e.g., iframes, embedded forms). Test to ensure they do not trap focus or require non-standard keyboard commands.

Examples

- Modals that trap focus without a close option.
- Dropdown menus requiring arrow keys but lacking a way to exit.
- Calendar widgets that do not allow navigation beyond the widget.

Exceptions

- Specific key sequences for security or specialized input

2.1.4 / WCAG 2.1 / Level A

Character Key Shortcuts

Users should be able to disable or remap keyboard shortcuts that involve a single key press.

Steps

1. Provide a way to turn off character key shortcuts.
2. Allow users to remap the shortcut to use one or more non-printable characters (e.g., Ctrl, Alt).
3. Ensure the keyboard shortcut for a component is only active when that component has focus.

Tips

- **Avoid Single Key Shortcuts as Defaults**. Include modifier keys like Ctrl, Alt, or Shift to prevent accidental activation.

Examples

- Pressing 'D' to delete items without a modifier key can result in accidental deletions.
- Pressing 'S' to save without focus can interrupt other tasks.

Exceptions

- Keyboard shortcuts with modifier keys (e.g., Ctrl+S).

2.2.1 / WCAG 2.0 / Level A

Timing Adjustable

Let users extend, adjust, or disable time limits on tasks.

Steps

1. Avoid time limits when possible.
2. Warn users before time expires and provide at least 20 seconds to extend the time limit.
3. Provide an option to extend the time limit by up to 10x the original limit.

Examples

- Website timeouts based on lack of activity
- Ecommerce checkouts that time out without warning
- Time limits on online forms (e.g., job applications, surveys)

Exceptions

- Real-time events (e.g., auctions)
- Time extension would invalidate the activity (e.g., security)
- Time limit is longer than 20 hours

2.2.2 / WCAG 2.0 / Level A

Pause, Stop, Hide

Let users pause, stop, or hide any content that is moving, blinking, scrolling, or auto-updating for more than 5 seconds.

Steps

1. Identify content that moves, blinks, or scrolls (e.g., animations, games, stock tickers).
2. Let users pause, stop, or hide moving content.

Exceptions

- Less than 5 seconds
- Moving content is essential (i.e., necessary for functionality or information)
- Triggered by user (e.g., user starts a video)
- Small, non-intrusive animations (e.g., loading spinner)

Examples

- Slideshow with a pause button
- Auto-updating news feeds can be disabled
- Stock ticker with stop control

2.3.1 / WCAG 2.0 / Level A

Three Flashes or Below Threshold

Nothing on a webpage flashes more than 3x per second.

Steps

1. Limit or eliminate any flashing content.

Tips

- **Flashing vs. Blinking**. Flashing can cause a seizure, while blinking can be distracting. Blinking becomes flashing if it occurs faster than 3x per second.
- **Test Flash Rates**. Use the Photosensitive Epilepsy Analysis Tool (PEAT).
- **Reduce the Contrast Ratio**. Especially with 3 flashes in one second.

Examples

- A gentle pulsing button that flashes less than 3x per second

Exceptions

- Emergency alerts
- Triggered by user
- Less than 3x per second

2.4.1 / WCAG 2.0 / Level A

Bypass Blocks

Enable users bypass repetitive blocks of content and skip directly to the main content.

Steps

1. Provide a “Skip to Main Content” link that allows users to skip repetitive blocks of content like the header and navigate directly to the main content..

Tips

- **Add Landmarks**. Provide alternative means of skipping content using landmarks. Use either semantic HTML (e.g., <nav>, <main>, <footer>) or ARIA landmarks (e.g., role="navigation", role="main"). Landmarks alone technically satisfy this criterion, but it’s recommended to have both in place.
- **Headings**. Implementing headings at the beginning of each section of content also technically satisfies this criterion. Again, multiple mechanisms are recommended.

Examples

- Allow users to skip repetitive headers, navigation menus, or advertisements to quickly reach the main article or content area.

Exceptions

- Simple pages with minimal or no repetitive content.
- Single-page applications (SPAs) where dynamic content eliminates the need for skipping repetitive sections.

2.4.2 / WCAG 2.0 / Level A

Page Titled

Each webpage has a unique and descriptive title that lets the user know the topic or purpose.

Steps

1. Ensure each web page has a unique and descriptive title that communicates its topic or purpose clearly.

Tips

- **Page titles should align with H1s**. Your page titles and H1s do not have to be the exact same, but they should be very similar.
- **Avoid Duplicate Titles**. No two pages should have the same title tag. For example, bank statement pages could differ by date: One title could include “Oct” and another title could include “Sept.”

Examples

- “Apple 15 Pro Specs”
- “How to Bake Gluten-Free Bread – Step-by-Step Guide”
- “About Our Company | Acme”

2.4.3 / WCAG 2.0 / Level A

Focus Order

Users can tab through your website using a keyboard in a logical order that that doesn't affect the meaning or operability.

Steps

1. Arrange interactive elements so that the focus sequence follows a logical and intuitive path for reading and interaction.
2. Align the programmatic focus order to match the visual order where possible. If the programmatic order differs, ensure it follows a logical, predictable sequence that users can follow intuitively.
3. Avoid using tabindex values greater than 0 to manually control focus order, unless absolutely necessary.

Tips

- **Tabindex**. Use tabindex="0" to include elements in the natural tab order and tabindex="-1" to remove elements from the tab order but allow programmatic focus. Avoid using positive tabindex values to prevent disrupting the expected keyboard navigation flow.

Examples

- A login form where the focus moves from username to password
- A modal dialog that traps focus within the dialog until it is closed.
- A multi-step form where the focus moves sequentially through the steps without skipping any required fields.
- A navigation menu where the focus moves sequentially through each menu item from top to bottom or left to right, depending on the layout.

2.4.4 / WCAG 2.0 / Level A

Link Purpose (in Context)

Users should know the purpose or destination of a link from the link text alone.

Steps

1. Write obvious anchor text in links (e.g., this is an example of **anchor text** but without a link).
2. Use descriptive alt text for linked images that clearly conveys the purpose or destination of the link..

Tips

- **Use Descriptive Text**. Avoid “click here” or “learn more.”
- **Avoid Long Anchor Text**. Don’t link a full sentence or a long URL (e.g., Amazon product URL).
- **Use Consistent Descriptions For The Same Destination**. Entirely different text implies a different page.

Examples

- Instead of “Click here,” say “Download Kris’s WCAG Guide.”
- Instead of “Learn more,” say “Learn more about WCAG.”

Exceptions

- Deliberately ambiguous links (e.g., game or quiz with hidden answer)
- Links that rely on surrounding context to provide meaning (e.g., to find out more about the new batch of strawberries **click here** – where click here is linked and the full context is clear**).**

2.4.5 / WCAG 2.0 / Level AA

Multiple Ways

Provide more than one way to find pages within your website.

Steps

1. Add multiple consistent ways to find pages (e.g., navigation menus, global search, sitemap, related pages links below content, breadcrumbs). Ensure these mechanisms are available and predictable on all relevant pages.

Tips

- **Add Search**. Use a search feature to allow users to easily locate content.
- **Add Breadcrumbs**. Provide links that show the path of a particular page within a website so that users know where they are located. Breadcrumbs aren't essential, but they can be helpful for websites with deeply nested pages.

Examples

- For an e-commerce site, provide global navigation for product categories, a search bar for specific items, and a sitemap for overall site structure.

Exceptions

- Pages within a sequential process (e.g., review cart page in checkout sequence).

2.4.6 / WCAG 2.0 / Level AA

Headings and Labels

When headings and labels are present, they adequately describe the topic or purpose of the following content.

Steps

1. Use clear, concise language for headings and labels, ensuring they accurately describe the topic or purpose of the associated content or functionality.

Tips

- **Not A Requirement**. This success criterion does not require headings or labels. Rather, the requirement is that only when headings or labels exist, they are descriptive.
- **What To Expect**. Headings and labels should match the content or function they describe to provide users with a clear and consistent understanding of what to expect.
- **Concise**. Headings and labels do not need to be lengthy, just descriptive. A single word can suffice.

Examples

- A label for a form field is “First Name”.
- An article about basketball statistics has a subheading of “Rebounds”.

2.4.7 / WCAG 2.0 / Level AA

Focus Visible

A clear visual indicator should display when interactive elements (e.g., links, buttons, form fields) receive focus.

Steps

1. Ensure interactive elements have a clear and consistent visual indicator (e.g., rectangle outline) that is visible when focused, regardless of browser or user settings..

Tips

- **Use a Contrasting Color**. Ensure the indicator stands out from the background and/or surrounding content.
- **Adapt Colors**. You may need to change the focus indicator color depending on the background to ensure visibility across different sections of your site.

Examples

- A button shows a blue outline when selected using the Tab key.
- A link is highlighted with a yellow background when focused.
- A form field displays a bold border when navigated to via keyboard.
- A text field with a vertical bar that indicates text can be added.
- A custom slider shows a thick outline or color change when adjusted using the keyboard.

Exceptions

- Elements that are not focusable or do not receive keyboard input.

2.4.11 WCAG 2.2 / Level AA

Focus Not Obscured (Minimum)

Focus should never be completely hidden by other content.

Steps

1. Ensure that focus indicators are never blocked by pop-ups, chatbots, sticky headers, etc.

Examples

- Active buttons that show an outline when focused
- Links that show a dotted underline when focused

Exceptions

- Disabled elements
- Triggered by the user
- Partial visibility is sufficient if users can easily reposition or scroll to fully view the focused element without losing focus.

2.5.1 / WCAG 2.1 / Level A

Pointer Gestures

Users should be able to perform all functionality with a single point of contact on a screen (e.g., simple taps or clicks rather than complex gestures).

Steps

1. Ensure that all actions can be performed with a single touch: single-click (or tap), double-click (or tap), long press, click and hold.

Tips

- **Avoid Path-Based Gestures**. These are gestures requiring specific patterns or shapes (like drawing a 'Z' or circle). Simple linear movements like dragging, swiping, or sliding are fine as long as single-point alternatives exist.
- **Avoid Multipoint Gestures**. If you use gestures requiring multiple contact points (e.g., pinch-to-zoom), provide a single-point alternative like a zoom button or slider.

Examples

- A photo gallery that allows both swipe navigation and next/previous buttons
- A map that offers both pinch-to-zoom and zoom in/out buttons
- A slider that can be adjusted by either dragging or clicking plus/minus buttons

Exceptions

- Simple linear movements (dragging, sliding, swiping) are allowed when alternatives exist
- Free-form drawing or handwriting gestures

2.5.2 / WCAG 2.1 / Level A

Pointer Cancellation

Let users cancel or undo actions from accidental or unwanted clicks.

Steps

1. Use up-events (release) rather than down-events (press) to trigger actions
2. If using down-events, provide at least one of these:
    - A way to undo the action
    - A way to abort before completion
    - A way to confirm before finalizing

Tips

- **Down Event vs. Up Event**. A down-event is when you press or click; an up-event is when you release. Up-events are safer because users can move away to cancel.
- **Essential Functions**: Only use down-events when absolutely necessary (like playing piano keys in a music app).

Examples

- A Delete button that only activates when released, not when first pressed
- A user who accidentally presses a button can slide away before releasing to cancel
- A submission form that asks "Are you sure?" before processing

Exceptions

- Games and similar interactions where down-events are essential
- Actions triggered by long presses

2.5.3 / WCAG 2.1 / Level A

Label in Name

Visual labels should match programmatic labels.

Steps

1. Ensure visible text is included in the accessible name (HTML or ARIA label)
2. Start the accessible name with the visible text when possible

Tips

- **Why Does It Matter?** Voice control users say what they see to activate controls. If the visual and code labels don't match, voice commands won't work.
- **Check All Text Labels**: This applies to buttons, links, form fields, and any interactive element with visible text.

Examples

- A button that says “buy” shouldn’t say “order” in the code.
- A search button showing "Search" should have "Search" in its accessible name, not "Find".

Exceptions

- Additional descriptive text can follow the visible text (e.g., visible: "Search" / accessible name: "Search product catalog").

2.5.4 / WCAG 2.1 / Level A

Motion Actuation

Provide alternative methods for actions that are initiated by motion (e.g., tilting, shaking).

Steps

1. Supplement motion triggers with key presses, button clicks, voice commands, or simple actions.
2. Let users disable moving or gesture activation.
3. Respond to platform/system motion settings

Tips

- **Ultimate Goal.** Users shouldn't need to move a device.

Examples

- Tilt-to-refresh with a button click.
- Shake-to-undo with a keyboard shortcut.
- Device rotation with a voice command.

Exceptions

- Motion-based games
- Motion that is being detected by motion/orientation sensors

2.5.7 / WCAG 2.2 / Level AA

Dragging Movements

Provide alternative methods for actions that are initiated by dragging.

Steps

1. Ensure that all actions can be performed with a single point of contact (e.g., tapping, clicking).
2. Include single pointer alternatives like buttons or menus

Tips

- **Think Point-and-Click.** Every drag action should have a simple click or tap alternative

Examples

- Resizing elements can be done with arrow keys.
- Dragging list items can be done with “up” or “down” buttons.
- Drag-and-drop file upload also has a "Choose File" button.
- Creating shapes can be accomplished by selecting predefined shapes.

Exceptions

- Games
- Specialized design tools

2.5.8 / WCAG 2.2 / Level AA

Target Size (Minimum)

Interactive elements should be at least 24 pixels by 24 pixels.

Steps

1. Design interactive elements that are large enough to be easily touched with a cursor, finger, or stylus

Tips

- **Add Padding.** Leave enough space between interactive elements to prevent accidental activation.

Examples

- Common touch targets: buttons, checkboxes, and form fields are all at least 24x24 pixels
- Social media icons aren't flush against one another and have sufficient space between one another

Exceptions

- The surrounding space can be included in the 24 x 24 metric
- Actions can be performed in alternative ways
- Target size was determined by the user
- Text link in a sentence or list
- Specificity is important (e.g., pin on map)

3.1.1 / WCAG 2.0 / Level A

Language of Page

Set a default language for each page to help users with screen readers (e.g., text-to-speech accuracy).

Steps

1. Add a lang attribute in the <html> tag of each page.

Tips

- **Check Source Cod**e. Type view-source: https://yourwebsite.com into a browser, then check the <html> tag for a *lang* attribute.

Examples

- **English**: <html lang="en">
- **French**: <html lang="fr">

Exceptions

- Symbolic content (e.g., math formulas)

3.1.2 / WCAG 2.0 / Level AA

Language of Parts

Content should specify when the language differs from the intended language.

Steps

1. Adjust the *lang* attribute in HTML to indicate when a language changes (i.e., when content should be read in another language).

Tips

- **Reading Directionality**. It's especially important when switching languages that read from left to right (vs. right to left).

Examples

- A paragraph in French on an English page should be marked with the *lang* Attribute (lang="fr")

Exceptions

- Common words that originated from a foreign language (e.g., siesta, hertz, homo sapiens)

3.2.1 / WCAG 2.0 / Level A

On Focus

Elements that receive focus shouldn't automatically trigger major changes like form submission or page navigation.

Steps

1. Provide controls and indicators so that users know when changes will occur.

Tips

- **Focus Doesn’t Mean Action**. Focus should only highlight an element, not trigger actions that change content or navigation.

Examples

- Don’t submit forms on button focus.
- Don’t trigger popups on focus.
- Don’t expand or collapse content on focus.

Exceptions

- Critical notifications
- Accessibility features that allow users to opt in or out of automatic changes

3.2.2 / WCAG 2.0 / Level A

On Input

Changing form fields, selections, or settings shouldn't automatically trigger significant changes without warning.

Steps

1. Ensure that nothing happens automatically when you type something into input fields (besides revealing the text).

Tips

- **Provide Controls**. Let users confirm before making significant changes based on their input.
- **Why No Auto-Changes?** Some users don’t easily perceive changes or are easily distracted by changes.

Examples

- Forms don't auto-submit when all fields are filled.
- Form errors don’t appear immediately upon keystrokes.
- Search results don’t update instantly upon keystrokes.
- Selecting a checkbox next to a shipping address doesn’t automatically complete the checkout.
- Changing a language from English to Spanish doesn’t reload the page.

Exceptions

- Users are informed beforehand

3.2.3 / WCAG 2.0 / Level AA

Consistent Navigation

Keep navigation elements in the same relative location and order across all pages within a set of pages.

Steps

1. Browse pages to ensure that common elements (e.g., search bars, skip navigation links) appear consistently.
2. Verify that repeated elements maintain the same order (e.g., menu items don't randomly reorder).

Tips

- **Think Relative**. Navigation usually follows the same order, but it doesn't necessarily have to be. What's important is that the same relative order remains intact.
- **Use Style Guides**. Design systems can help maintain uniformity.

Examples

- Navigation bar remains at the top of each page.
- Footer with contact information on every page.
- Sidebar menus with the same categories and links.

Exceptions

- Users initiate a change
- Context-specific navigation (e.g., multi-step form)

3.2.4 / WCAG 2.0 / Level AA

Consistent Identification

Elements with the same functionality should have consistent labels, names, and text alternatives.

Steps

1. Review all repeated functionality across your site to ensure consistent labels (e.g., don't use "search" in one place and "find" in another).

Tips

- **Consistent Isn't Identical**. You could label the same button with "Go to Page 4" then "Go to Page 5."
- **Labels vs. Functions**: In 2.5.3, we explained how visual labels should match their programmatic labels, here what matters is that the same function uses the same identification across your site.

Examples

- Facebook share icons with the same alt text.
- Search functionality labeled as "search" consistently throughout the site, not "find" or "query" in different places.

Exceptions

- Similar elements that serve a different purpose (e.g., the same checkmark icon could say "correct," "finished," or "approved")

3.2.6 / WCAG 2.2 / Level A

Consistent Help

Help and support options should be in consistent locations across pages that use the same layout.

Steps

1. Place help options in the same location. If a support link is in the bottom-right of a page, it should be in the bottom-right of other pages.

Tips

- **Help Isn't Required**. This criterion is about consistent help.
- **Help Features**: Help includes human contact, self-help, and automated support options.

Examples

- Contact information (e.g., phone number, email, hours)
- Methods of contact (e.g., messaging, chat box, contact form, social media)
- Self-help: FAQ, knowledgebase, tool tips with consistent placements and designs

Exceptions

- Pages that serve different audiences or purposes (e.g., help during product pages could differ from help during checkout pages). But you still want to make accessing help as easy and as intuitive as possible.

3.3.1 / WCAG 2.0 / Level A

Error Identification

Notify users when they make errors while entering input (and how to fix them).

Steps

1. Clearly identify input errors in text. Adding a red border around input fields with an error is a great visual indicator.
2. Highlight all fields with errors and summarize them at the top of a form.

Tips

- **Use ARIA**: Add aria-invalid="true" and use aria-describedby to link error messages with their respective fields.
- **Timing Matters**: Be strategic with validation timing to avoid overwhelming users with multiple errors.

Examples

- Incorrect email format (e.g., name@gmailcom)
- Improperly formatted date (e.g., entering 21 instead of 2021)
- Missing value in a required field (e.g., no address)

Exceptions

- This only applies to automatically detectable errors (e.g., invalid email address vs. name).

3.3.2 / WCAG 2.0 / Level A

Labels or Instructions

Instructions should be clear and descriptive so that users know how to interact with forms and other elements.

Steps

1. Assign a clear label (e.g., First Name) to every form field or interactive element.
2. Provide instructions for any special formatting requirements (e.g., date format, password rules).

Tips

- **Use Proper Labels**. Connect labels to form fields using the HTML <label> element with for attributes or ARIA labels.
- **Provide Examples**. What is the expected input format?
- **Mark Required Fields**. Use an asterisk or label.

Examples

- “Submit Application” is better than “Submit”
- “Enter your 10-digit phone number” is better than "Phone Number"

Exceptions

- Instructions can be provided through alternative means when standard labels don't fit the context (e.g., complex or dynamic elements like grid headers or interactive widgets).

3.3.3 / WCAG 2.0 / Level AA

Error Suggestion

Provide suggestions to fix errors.

Steps

1. Provide helpful error messages (e.g., “Please enter a valid email address”).
2. Tell users exactly what's wrong and how to fix it (e.g., "Email must contain an @ symbol" rather than just "Invalid email"),

Tips

- **Avoid Technical Jargon**. Communicate in plain words.
- **Preserve Form Data**. Don’t force users to re-enter data
- **Strategic Timing**: Consider when to show error messages - after field completion or form submission.
- **Beyond Identification**: While 3.3.1 identifies errors and 3.3.2 provides upfront instructions, this criterion requires specific suggestions for fixing errors after they occur.

Examples

- “Zip Code is required”
- “Password must be at least 8 characters long.”
- “Did you mean January?”

Exceptions

- Suggestions that jeopardize security (e.g., don’t reveal whether an account exists after users enter an email).

3.3.4 / WCAG 2.0 / Level AA

Error Prevention (Legal, Financial, Data)

For submissions involving legal commitments, financial transactions, or data modifications, ensure actions can be reviewed or reversed.

Steps

1. Check the format and accuracy of input before saving.
2. Let users review, confirm, and correct information before submitting.
3. Add the ability to reverse actions (e.g., cancel, undo).

Tips

- **Add Confirmation Dialogs**. Let users review and confirm the input.
- **Critical Actions**: This applies to legal commitments (contracts), financial transactions (purchases), or data changes (deleting account data).

Examples

- A user adds an extra “0” resulting in a $2,500 transaction instead of $250.
- Selecting a hotel reservation for “September” instead of “August”.
- A user quickly tries to dismiss a pop-up and unwittingly accepts the terms of service.

Exceptions

- Minor errors

3.3.7 / WCAG 2.2 / Level A

Redundant Entry

Help users avoid re-entering the same information multiple times.

Steps

1. Whenever the same information is required again, autofill this input or let users select from previous input.

Tips

- **User Control**. Always let users choose whether to reuse their information.

Examples

- A checkbox that copies a billing address to a shipping address.
- Reusing previously entered billing information during checkout.
- Auto-suggesting previously used email addresses in a form.

Exceptions

- Repetition is critical (e.g., password).
- Previous information is no longer valid (e.g., code expired).

3.3.8 / WCAG 2.2 / Level AA

Accessible Authentication (Minimum)

Users can log in without relying on cognitive functions (e.g., remembering a password).

Steps

1. Provide an alternative login method.
2. Allow users to copy/paste passwords and use password managers
3. Allow users to bypass cognitive tests.

Tips

- **Cognitive Function**. Covers memorization, transcription, and mental processing.
- **Cognitive Tests**. Involve recognizing objects or non-text content (e.g., images, video, audio).

Examples

- Password managers that remember passwords for users.
- The ability to copy and paste to reduce the burden of typing.
- Authentication by device, QR codes, or OAuth.

Exceptions

- Systems that don't require authentication.
- Strict authentication is required for legal compliance.
- Security is compromised by alternative methods.

4.1.1 / WCAG 2.0 / Level AA

Parsing (Obsolete and Removed in WCAG 2.2)

All elements in markup languages can be successfully parsed by assistive technologies.

Steps

1. All elements should have an opening and closing tag.
2. All elements should be nested correctly.
3. All element IDs and attributes should be unique.
4. Ensure no broken links exist.

Tips

- **Validate Your Code**. Use tools like the W3C Markup Validation Service to search for code errors.

Examples

- Semantic HTML tags like <header>, <nav>, <main>, and <footer>.

4.1.2 / WCAG 2.0 / Level A

Name, Role, Value

For all interactive elements, users should know the name (what it is), role (what it does), and value (current state).

Steps

1. Use semantic HTML elements and/or ARIA attributes to define the name, role, and value.

Tips

- **Check 3rd Party Code**. Review plugins, widgets, or scripts.
- **Names Come From Many Sources**: Names can be visible labels (including adjacent text), aria-label, aria-labelledby, or alt text.
- **Values Should Match the Current Input**. Changing a quantity from 1 to 4 should update the value.
- **Properties and States Are Values**: Values include both stable properties (e.g., "readonly", "required") and changing states (e.g., "checked/unchecked", "expanded/collapsed"). Both must be communicated to assistive technology.

Examples

- **Name**: A submit button announces “Submit button” on screen readers.
- **Role**: A checkbox is identified as a checkbox, not generic input.
- **Value**: A slider announces the current value of “Volume 50%.”
- **State**: A menu announces "expanded" or "collapsed."
- **Properties**: A form field announces "required" or "readonly" when focused.

Exceptions

- Decorative or non-interactive elements

4.1.3 / WCAG 2.1 / Level AA

Status Messages

Ensure that users are aware of important messages (e.g., errors, updates).

Steps

1. Add ARIA live regions or status roles to ensure messages are announced in screen readers without disrupting focus.

Tips

- **How ARIA Live Works**. When content changes (like removing a cart item), use aria-live to ensure screen readers announce the update without disrupting focus. The message appears visually and is announced automatically.
- **Choose the Right Level**: Use "polite" for most updates and "assertive" only for critical messages users need to know immediately.

Examples

- Error messages displayed after form submission.
- Notifications of successful form submission.
- Alerts about invalid data entered in a form field.
- Confirmation messages after user actions like deleting an item

Exceptions

- Error messages in dialogs (since they have entered focus)

Next Steps

WCAG Course

Would you like to save 20+ hours and learn WCAG from me?

My WCAG Course helps you master WCAG with:

- Video explanations of all success criteria
- Code examples
- Ready-to-use checklists
- Role-based accessibility workflows

ADA Compliance Course

Need to reduce your ADA lawsuit risk quickly?

My ADA Compliance Course shows you exactly which issues to fix first.

YouTube

My Want free accessibility and compliance knowledge? Visit @adabook on YouTube for practical guides on audits, VPATs, WCAG, certification, training.

Accessible.org

Ready for expert help? Our audit and VPAT/ACR services offer transparent pricing and fast turnaround. Visit Accessible.org to get started.