

What's New in the Web Content Accessibility Guidelines (WCAG) 2.2

Lainie Strange
OA-ITSD, Web Accessibility Specialist

Agenda

- ▶ Brief History of the Web Content Accessibility Guidelines (WCAG) and State Adoption
- ▶ Six A/AA WCAG 2.2 Guidelines
 - ▶ 2.4.11 Focus Not Obscured (Minimum) (AA)
 - ▶ 2.5.7 Dragging Movements (AA)
 - ▶ 2.5.8 Target Size (Minimum) (AA)
 - ▶ 3.2.6 Consistent Help (A)
 - ▶ 3.3.7 Redundant Entry (A)
 - ▶ 3.3.8 Accessible Authentication (Minimum) (AA)

Brief History of WCAG and the State Standard

WCAG Version	Publication Date	Focus	MO State Adoption
1.0	1999	HTML accessibility	2003
2.0	2008	Four core POUR principles	2017
2.1	2018	Addresses mobile accessibility, low vision, cognitive and learning disabilities	2024
2.2	2023	Addresses mobile accessibility, low vision, cognitive and learning disabilities, especially on mobile devices	2025

- [RSMo 161.935](#) – Effective 1999, requires accessibility of Information and Communication Technology
- The State of Missouri began adopting the WCAG guidelines through the State's Accessibility Standard document

WCAG 3.0?

- ▶ Renamed: W3C Accessibility Guidelines
- ▶ First draft, 2016
- ▶ Currently a working draft
- ▶ September 2025 Update: Publication schedule will be announced early 2026
- ▶ The conformance model will change
- ▶ The state's current standard outlines that any new WCAG guidelines will be adopted **two years after the publication date**

2.4.11 Focus Not Obscured (Minimum) (AA)

When an element receives focus, it must be at least partially visible.

Role:

- ▶ Designer
- ▶ Developer

What is it?

Any item receiving keyboard focus should always be at least partially visible on the user's browser window. This guarantees that focused elements are not completely hidden by other content like sticky headers, footers, or modals. And, while it's best to keep the elements fully visible, it's not required at the AA level.

Who is Affected?

- ▶ People who rely on keyboards or assistive technology to navigate.
- ▶ People with low vision.
- ▶ People with cognitive disabilities.



2.4.11 Focus Not Obscured (Minimum) (AA)

Why does it matter?

- ▶ When focused elements are hidden or blocked from view, users relying on keyboards or assistive technology can't tell where their interaction is happening. This makes it harder to navigate and complete tasks, leading to frustration. Hidden elements can also make it seem like the system isn't responding.

2.4.11 Focus Not Obscured (Minimum) (AA)

Testing Focus Visibility

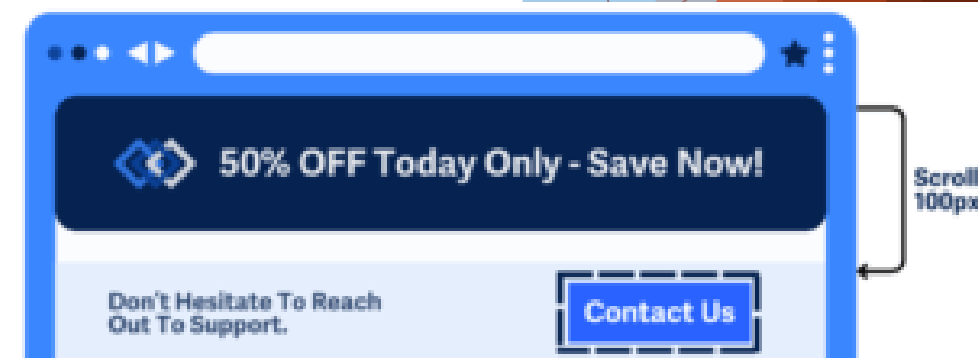
- ▶ As users tab through a page, all focused elements should remain at least partially visible. Make sure other content, such as modals, banners, or dialogs, doesn't fully block focused elements.
- ▶ Design modal dialogs to take focus when they appear and prevent interaction with underlying content. Ensure notifications like cookie banners are dismissible, preventing them from obscuring focusable elements.
- ▶ Check content magnified up to 200%, to ensure the resized layout doesn't hide interactive elements. components like sticky headers or footers.
- ▶ In this example, the scroll-padding-top and scroll-padding-bottom values adjust the page layout to ensure interactive elements, like form fields or buttons, are visible when focused.

```
html {  
  scroll-padding-top: 100px; /* Space for a sticky header */  
  scroll-padding-bottom: 50px; /* Space for a sticky footer */  
}
```



- ▶ As a bonus, the scroll-behavior property helps create smooth scrolling transitions, so the sudden shift isn't too harsh.

```
/* Example usage for smooth keyboard navigation */  
body {  
  scroll-behavior: smooth;  
}
```



2.5.7 Dragging Movement (AA)

Actions that require dragging (like reordering) must also be possible using buttons or another method that does not require dragging.

Roles Affected:

- ▶ Designer
- ▶ Developer

What is it?

- ▶ If a task on your website requires dragging, there must be an easier alternative provided.
- ▶ Instead of dragging, users should be able to complete the same action with a single tap, click, or other simple input.
- ▶ Dragging motions are typically found in things like sliders, color pickers, sortable lists, Kanban boards (where you drag and drop tasks), and interactive maps.
- ▶ While this sounds a lot like **WCAG 2.5.1 Pointer Gestures**, they aren't the same: with dragging, only the start and end points matter, but gestures usually mean the user has to follow a path with their mouse or finger.
- ▶ Similarly, the **WCAG 2.1.1 Keyboard** success criteria require that dragging movements be keyboard accessible. However, that's not enough to meet this one. Dragging must also work with a single tap or click for users who don't use keyboards.

2.5.7 Dragging Movement (AA)

Why does it matter?

- ▶ Without single-pointer options for dragging, some users might not be able to use certain parts of a website. People with mobility issues might find dragging difficult or tiring, like when panning across a map or dragging items in a website builder.
- ▶ People who use assistive tools like trackballs or eye-gaze systems might not be able to make dragging motions at all. Simple features like sliders for volume control can be completely inaccessible.

Who is affected?

- ▶ People with mobility or fine motor control issues
- ▶ People using different types of input devices

2.5.7 Dragging Movement (AA)

How to implement 2.5.7

Single Pointer Alternatives

- ▶ For any component, element, or interface that needs dragging, make sure there's an easy single-pointer option as an alternative.
- ▶ Single-pointer options can include clicking a button or typing in a value. For example:
 - ▶ For a volume slider, allow users to type a number value into an input or click simple +/- buttons to adjust the volume
 - ▶ For sortable lists, allow users to reorder items with up/down buttons
 - ▶ For interactive maps, allow users to pan using directional buttons
- ▶ Remember, while offering keyboard shortcuts as alternatives to dragging is great and necessary for other success criteria, it's not sufficient to meet this success criterion. You need to provide a single-click/tap alternative or a simple input.



2.5.8 Target Size (Minimum) (AA)

Targets must be at least 24×24px, unless they are part of a sentence or block of text, surrounded by enough space, or near another target with the same function that meets the size.

Role Affected:

- ▶ Designers
- ▶ Developers

What is it?

- ▶ Buttons, links, and other interactive elements should be at least 24 x 24 pixels. If that's not possible, they should have enough empty space around them to make clicking easier.
- ▶ Even if an element is smaller than 24 x 24 pixels, it can still pass if there's enough space between it and other clickable elements. But it's always a good idea to make them bigger for easier use.

2.5.8 Target Size (Minimum) (AA)

Why does it matter?

- ▶ Tiny buttons and links are frustrating to click or tap, especially on touchscreens like phones and tablets. When they're placed too close together, it's even easier to tap the wrong one by mistake.
- ▶ People with limited hand control—like those with tremors, arthritis, or other mobility challenges—may struggle to tap small buttons. If elements are too close together, they could tap or click the wrong one or not be able to use it at all.

Who is affected?

- ▶ People with mobility impairments.
- ▶ People on smaller screens.
- ▶ People in unstable environments.
- ▶ People with large fingers.

2.5.8 Target Size (Minimum) (AA)

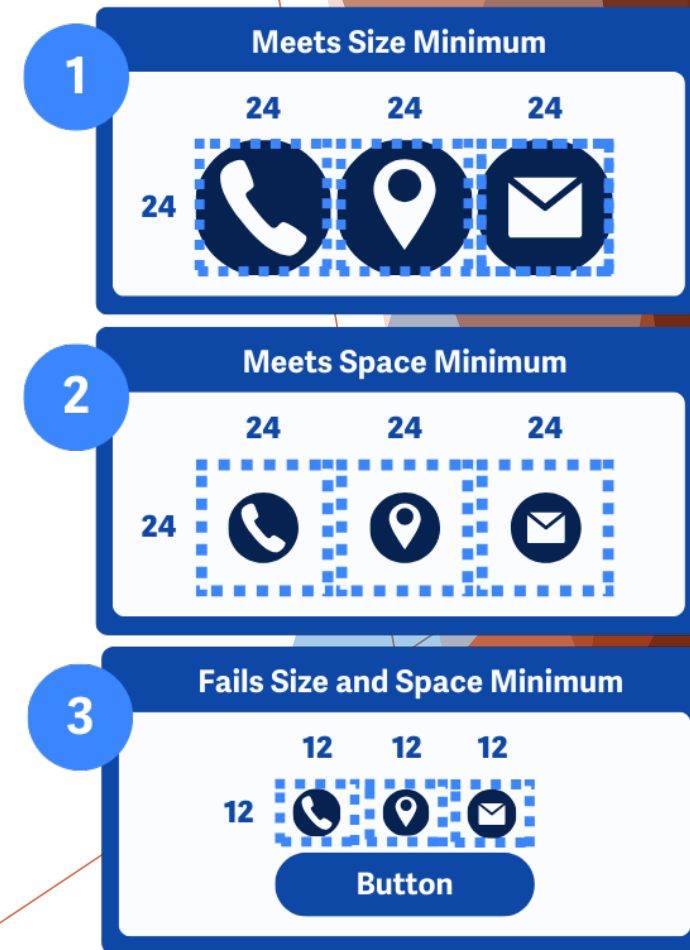
How to implement 2.5.8 - Minimum Size and Spacing

Make all buttons, links, and interactive elements at least 24 x 24 pixels. If that's not possible, add extra space around them so they're easier to tap.

To make sure buttons meet the size requirement, you can adjust CSS settings. Use the min-height and min-width CSS properties to set the minimum size. Or, add padding to help give them a total size of 24 x 24 with the extra spacing in between.

In the image to the right, there are examples of passing and failing adjacent icons in image:

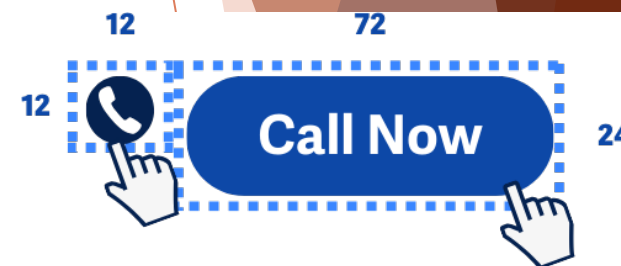
- ▶ In example 1, the buttons are at least 24 x 24 pixels, so they meet the rule.
- ▶ In example 2, the buttons are smaller, but because they have enough empty space around them, they still pass.
- ▶ In example 3, the buttons are both too small and too close together, making them hard to tap. They don't meet the rule.



2.5.8 Target Size (Minimum) (AA)

How to implement 2.5.8 - Equivalent Action Element

- ▶ An exception to this rule is if there's another element with the equivalent action on the page. If this is true, the element can keep its size smaller than 24 x 24 pixels. Note that other elements must meet this success criterion.
- ▶ To the right is an example of a small interactive element to initial a call right next to a much larger button element with the equivalent action of calling.



Exception for Inline Elements

- ▶ Interactive Elements inline with text, such as sentences, paragraphs, etc., are an exception since the line height limits the size, and it's impossible to tell where the interactive element will fall as screen sizes change and the text reflows.
- ▶ However, it's a good idea to increase the line height to keep the text readable and easy to tap or click.
- ▶ Below is an example of a paragraph containing three links, each line is only 12 pixels high, so the links would technically fail the success criterion. But, since they are part of inline text, they are an exception. However, it's recommended that the line height be increased.

Check out the most popular software tools: [Mock & Roll - a mockup creation tool](#), [Fontastic! - A typography resource](#), and [Hue Kidding Me? - A color palette generator](#).

12 pixels
12
12

2.5.8 Target Size (Minimum) (AA)

Exception for Essentials

- ▶ Lastly, if it's absolutely necessary or legally required for the target size and spacing to be smaller than 24 x 24 pixels, this rule can be skipped.
- ▶ Below is an example of a map where the pins are too close together and not big enough, but it's necessary since we can't control where the pins will be placed, and if they are too large, they will be indistinguishable from each other.



3.2.6 Consistent Help (A)

Help options (like contact link, support widget) must appear in the same place across pages.

Roles Affected:

- ▶ Designers
- ▶ Developers
- ▶ Content Creators/Editors

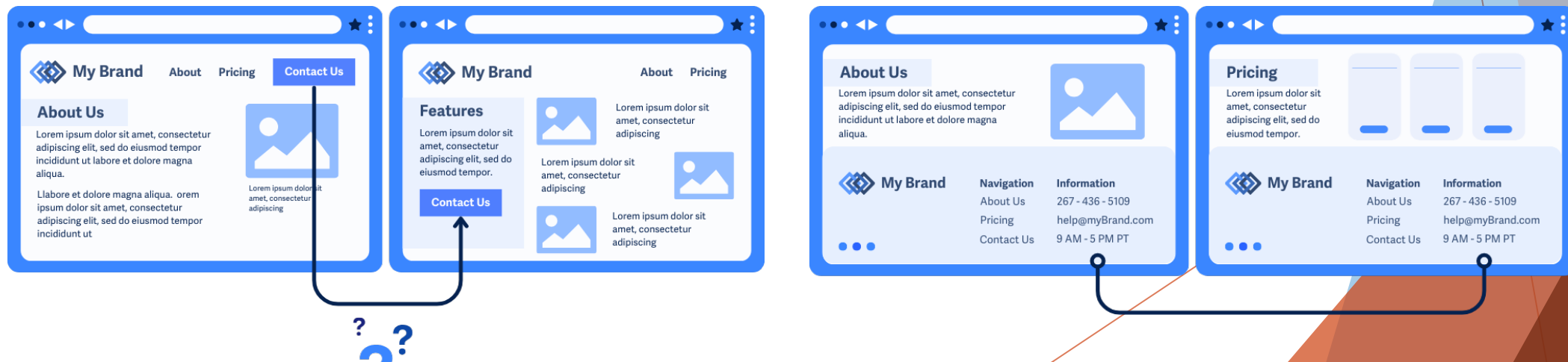
What is it?

- ▶ If a website offers options or info to help users with questions or issues, their placement on relevant pages should be consistent and stay in the same general order compared to the other elements on the page. Examples:
 - ▶ Contact forms or chat systems to reach human support teams
 - ▶ Chatbots to reach automated systems
 - ▶ Contact info such as phone number, email address, hours of operation
 - ▶ Self-help options such as documentation pages or frequently asked questions sections

3.2.6 Consistent Help (A)

Why does it matter?

- ▶ When help options or details are scattered across different pages, it can be tough for users to find the help they need. Moving around contact links, chatbot popups, or FAQ sections across similar pages can confuse users as they browse a site.
- ▶ For example, imagine seeing a “Contact Us” link in the header of the homepage. Then, you visit the features page and have a question. You’d expect the “Contact Us” link to still be in the header, but it’s now at the bottom of the sidebar instead. You might get frustrated and leave the page without asking your question.



3.2.6 Consistent Help (A)

Who is affected?

- ▶ People with cognitive disabilities.
- ▶ People with low or limited vision.
- ▶ People who are blind.

How to implement 3.2.6

- ▶ First, go through the pages on your site and figure out which groups of pages make up related sets. For example, agency divisions or high-level programs could be part of the same set. Or, a news release where each post uses a similar template would be considered a unique set of pages. You may also have sitewide contact information in a header, footer or side navigation.
- ▶ Next, find all the helpful options or information that fall under these categories (if any):
 - ▶ Contact Details
 - ▶ Contact forms or messaging systems
 - ▶ Self-help options, such as FAQ sections or links to a documentation page
 - ▶ Chatbots

3.3.7 Redundant Entry (A)

Don't ask for the same information twice in the same process. Provide pre-filled fields or selection options if the information was already given.

Roles Affected:

- ▶ Designer
- ▶ Developer

What is it?

- ▶ A web form should not ask users to enter the same information more than once in a single session. This is especially important for multi-step forms and processes. For example, when completing a checkout form, users should not have to separately enter their billing and shipping information if they are the same.

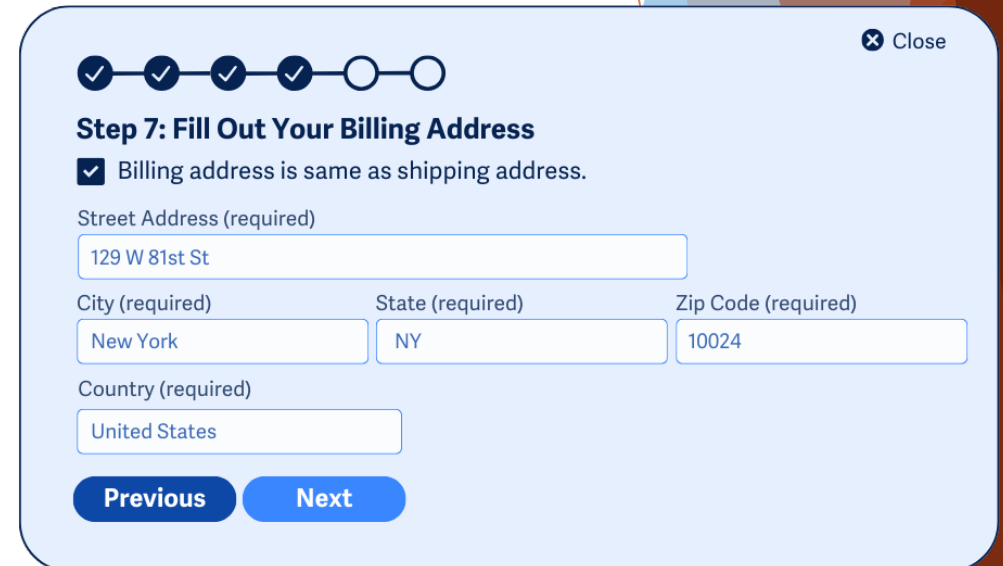
3.3.7 Redundant Entry (A)

Why does it matter?

- ▶ Requiring users to remember and re-enter information can cause stress and mistakes.
- ▶ Everyone experiences mental fatigue while completing multi-step processes and adding memory tasks makes the process even more difficult.
- ▶ Nobody enjoys re-entering the same information over and over.

Who is affected?

- ▶ People with cognitive disabilities.
- ▶ People with limited mobility.



A screenshot of a multi-step form interface. At the top, a progress bar shows six steps, with the first four marked with checkmarks and the last two with empty circles. The current step is labeled "Step 7: Fill Out Your Billing Address". Below this, there is a checkbox labeled "Billing address is same as shipping address." which is checked. The form contains several input fields: "Street Address (required)" with the value "129 W 81st St", "City (required)" with "New York", "State (required)" with "NY", "Zip Code (required)" with "10024", and "Country (required)" with "United States". At the bottom, there are two buttons: "Previous" and "Next". A "Close" button with an 'X' icon is in the top right corner.

Close

Step 7: Fill Out Your Billing Address

☒ Billing address is same as shipping address.

Street Address (required)

129 W 81st St

City (required) State (required) Zip Code (required)

New York NY 10024

Country (required)

United States

Previous Next

3.3.7 Redundant Entry (A)

How to implement 3.3.7

Avoid Duplicate Input

- ▶ Remove any fields that ask for the same information more than once.
- ▶ Make sure every form input is necessary

Populate Data From Previous Input

- ▶ Instead of making users remember what they entered before, auto-fill previous data
- ▶ For fields that appear more than once, make sure there's a way to populate them easily. This can be done in a number of ways:
 1. Automatically filling the field and letting the user edit if needed
 2. Providing a checkbox to let the user choose to fill the field with previous information
 3. Asking the user to confirm if the previous information is still correct. If not, showing new input fields

3.3.7 Redundant Entry (A)

Most of the time, redundant inputs should be avoided, but there are some **exceptions**, such as:

- ▶ Games where repeating inputs is part of the challenge, such as memory quizzes
- ▶ Confirming passwords or email addresses for security or validation reasons
- ▶ When previous entries expire and need to be updated

3.3.8 Accessible Authentication (Minimum) (AA)

Authentication must not rely on memory alone. Allow copy-paste, password managers, or other options (like email verification).

Roles Affected:

- ▶ Designer
- ▶ Developer

What is it?

- ▶ Authentication, or the ability to log into a website, should not feel like a memory test. Having to remember passwords, transcribe codes, or solve puzzles can make logging in unnecessarily difficult.
- ▶ The goal is to make authentication simple for everyone, especially those who have difficulty with memory or complex tasks.

Login Form

✉ Email Address

myEmail@example.com

myEmail@example.com

otherEmail@example.com

👤 Password

Aardvark42!Shhh

Login To Your Account

3.3.8 Accessible Authentication (Minimum) (AA)

Why does it matter?

Issues that can happen during inaccessible authentication setups:

- ▶ People forgetting their password and struggling with password resets
- ▶ Difficulty figuring out how to transcribe one-time or time-limited codes from another device
- ▶ Getting stuck on CAPTCHAs that require solving puzzles, math problems, or working with blurry images
- ▶ These issues cause frustration and can prevent people from accessing critical services like banking, healthcare, or work-related platforms.

Who is affected?

- ▶ People with cognitive disabilities.

3.3.8 Accessible Authentication (Minimum) (AA)

How to implement 3.3.8

1. **Identify Authentication Steps:** Locate all processes that require a user to authenticate, such as logging in, completing a security check like CAPTCHA, or resetting a password.
2. **Evaluate Cognitive Function Tests:** For each step, determine if a cognitive function test is required. This includes tasks such as:
 - ▶ Remembering a password or code.
 - ▶ Solving a puzzle.
 - ▶ Transcribing information from another source, like a code
 - ▶ Identifying non-text content provided by the user (e.g., a CAPTCHA with personal images).
 - ▶ Recognizing objects (e.g., selecting all images with a car).

3.3.8 Accessible Authentication (Minimum) (AA)

3. Check for Exceptions to the Cognitive Test Requirement:

- ▶ Is the test a cognitive function like object recognition or personal content identification? If so, the criterion is met.
- ▶ Is it a **general cognitive function test**? If the test requires memory or transcription (and is not object or personal content recognition), then an alternative or mechanism must be available.

3.3.8 Accessible Authentication (Minimum) (AA)

4. **Verify Alternatives or Mechanisms:** If a general cognitive function test is required, ensure **one** of the following is available:

An Alternative Authentication Method:

1. Biometric authentication (fingerprint, facial recognition).
2. Passwordless login (e.g., email magic links).
3. Other methods not reliant on cognitive ability

A Mechanism to Assist the User:

4. Allowing users to paste into password fields.
5. Providing support for password managers to autofill credentials.
6. Enabling copy and paste for security codes.

Login Form

✉ Email Address

myEmail@example.com

Send Login Link to Inbox



3.3.8 Accessible Authentication (Minimum) (AA)

5. Verify Input Field Behavior:

- ▶ Ensure that input fields do not prevent users from pasting the entire password or code in one action.
- ▶ Confirm that the structure of the fields does not make it impossible to use a password manager or copy the code.

Sending a code to a separate device

If the code is sent to a separate device, such as via SMS, then it's not necessary to validate whether users can copy or transfer the code between devices. Only the final input needs to be tested (meaning the final input allows copy/paste).

Questions?

◆ lainie.strange@oa.mo.gov

Resources

◆ [WCAG 2.0](#) and [2.1](#)

◆ [State of Missouri Information Communication Technology \(ICT\) Accessibility](#)

◆ [MO Assistive Technology YouTube Channel](#)

◆ [State Website and Applications Accessibility Testing Page \(including testing checklist for all WCAG 2.2 Guidelines\)](#)