

# But...why??

A look at common patterns found on the web,  
how they are harmful for accessibility, and what to do instead.

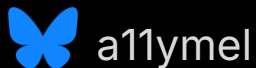
Melanie Sumner

Product Accessibility Lead, Design Systems  
HashiCorp




# Melanie Sumner

- Product Accessibility Lead, HashiCorp
- Invited Expert, WAI-ARIA & AccName, W3C
- Core Team, Ember.js



# Open source projects

Everyone is welcome to participate!

A graphic for the A11y Automation Tracker project. It features a blue robot character holding a magnifying glass and a clipboard, set against a dark background with a city skyline. The text 'A11y Automation Tracker' is prominently displayed in white and blue.


**A11y Automation Tracker**

A more thorough way to track the potential accessibility violations and the automated linters and tests currently available.

**A11Y AUTOMATION TRACKER**

Tracks availability of automated checks for potential a11y violations.

<https://a11y-automation.dev>

A graphic for the 'Please Fund A11y' project. It features a blue banner with the text 'ACTION ITEM' and 'VISIBLY SUPPORT ACCESSIBILITY EFFORTS' over a background of colorful graffiti, including the word 'GOOD' in large letters.


**ACTION ITEM**

**VISIBLY SUPPORT ACCESSIBILITY EFFORTS**

**PLEASE FUND A11Y**

This guide outlines the places where open-source accessibility work would benefit from funding.

<https://pleasefunda11y.com>

A graphic for the 'Learn to Dev' project. It features a blue banner with the text 'LEARN TO DEV' on a yellow background with a pattern of dots. Below the banner, it says 'LEARN HOW TO MAKE THINGS FOR THE WEB!'.

LEARNTODEV.INFO

**LEARN TO DEV**

LEARN HOW TO MAKE THINGS FOR THE WEB!

**LEARN TO DEV**

So you want to get into building things for the web? Here are some resources to get you started; you'll find websites, videos, podcasts, and other things!

<https://learntodev.info/>

# Other a11y talks

Continuous Accessibility:  
Strategies for Success  
at Scale

**SHIFT LEFT**  
*Melanie Sumner*

A designer and  
developer walk  
into a bar...



RETHINK REAL:  
ACCESSIBLE WEB  
DEVELOPMENT

**notist** melsumner

# So, why this talk?

Because...reasons.

- Well-defined process
- Semantic markup
- Beautiful design
- Accessibility non-negotiable
- Tests
- Thorough documentation

A dark-themed user interface mockup for the Helios Design System. It features a header bar with three dots, a main content area with the title 'Helios Design System', and a right-hand sidebar with various interactive elements like checkboxes, sliders, and a color palette. The text 'Helios' is in blue, and 'Design System' is in white.

## Helios Design System

The Helios Design System from HashiCorp provides the open source building blocks to design and implement consistent, accessible, and delightful product experiences.

[Release Notes](#) →

[Helios Roadmap](#) ↗

[Share Feedback](#) ↗

**accessible design system**  
**!=**  
**accessible application**

# Agenda

1. First, we'll talk through some UI examples.
2. Then, we'll talk about where they fail accessibility.
3. Then I will show you what to do instead.
4. P.S. We may have some detours
5. P.P.S. I will try very hard to fit this into the time limit



# Some Disclaimers

Progress, not perfection. And other things.

The code you are about to see may be real. Or made up. Or some combination thereof.

Any resemblance to code living or dead is purely coincidental...or not. It's not to shame anyone, but rather show real examples of UI and the real ways it fails accessibility. Because once we know, we can fix it. We can do better next time.

I've said it before and I'll say it again: accessibility is about progress, not perfection.

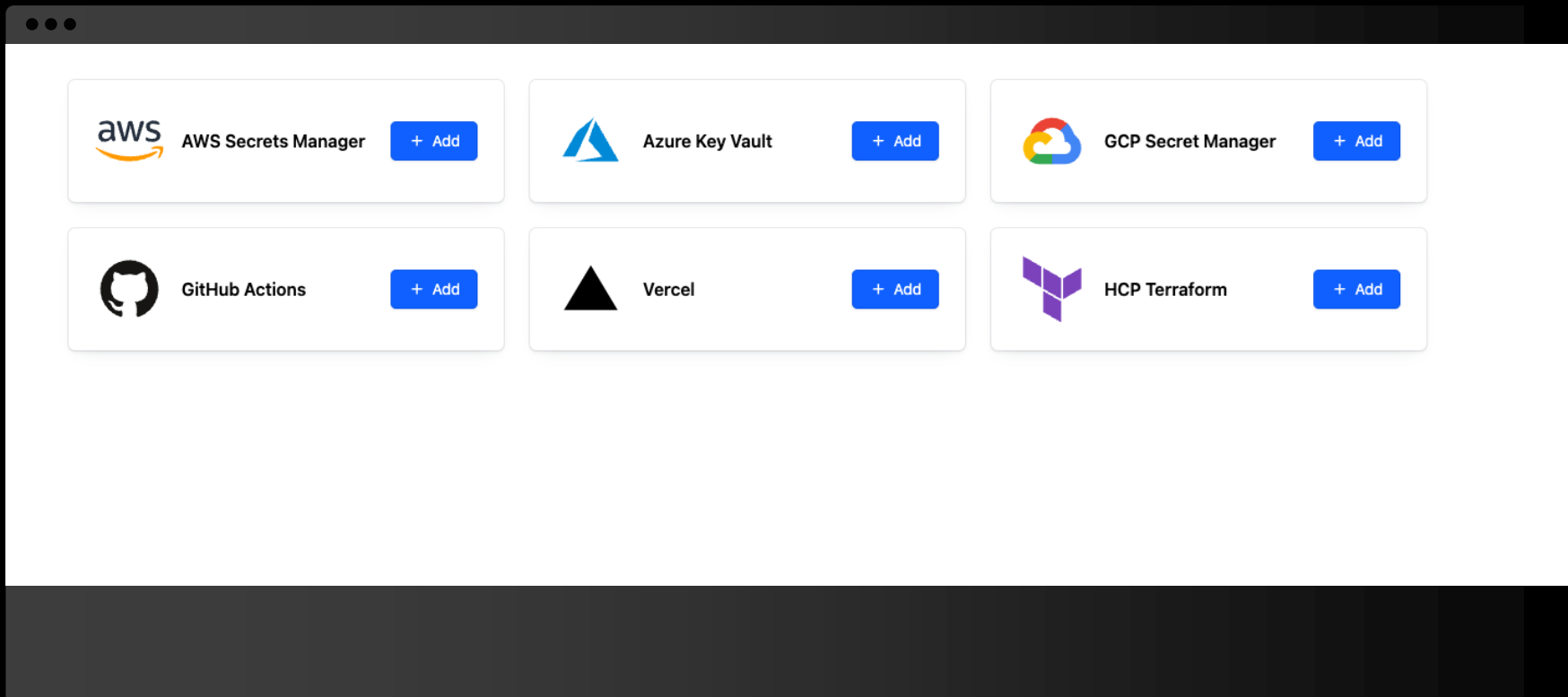
*Finally, all views expressed are my own and may not reflect the views of my employers, the web at large, or even other people who also work in the field of digital accessibility.*



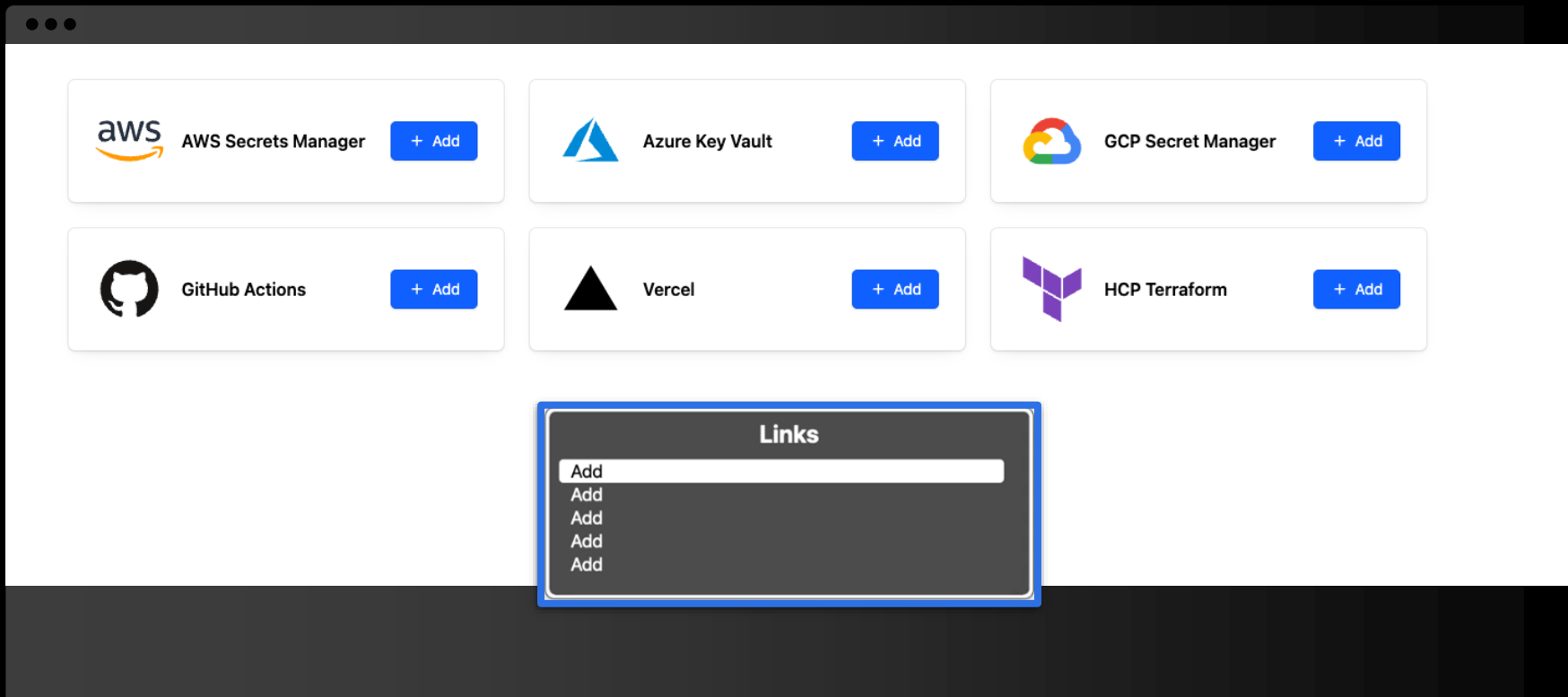
01

# Accessible Names

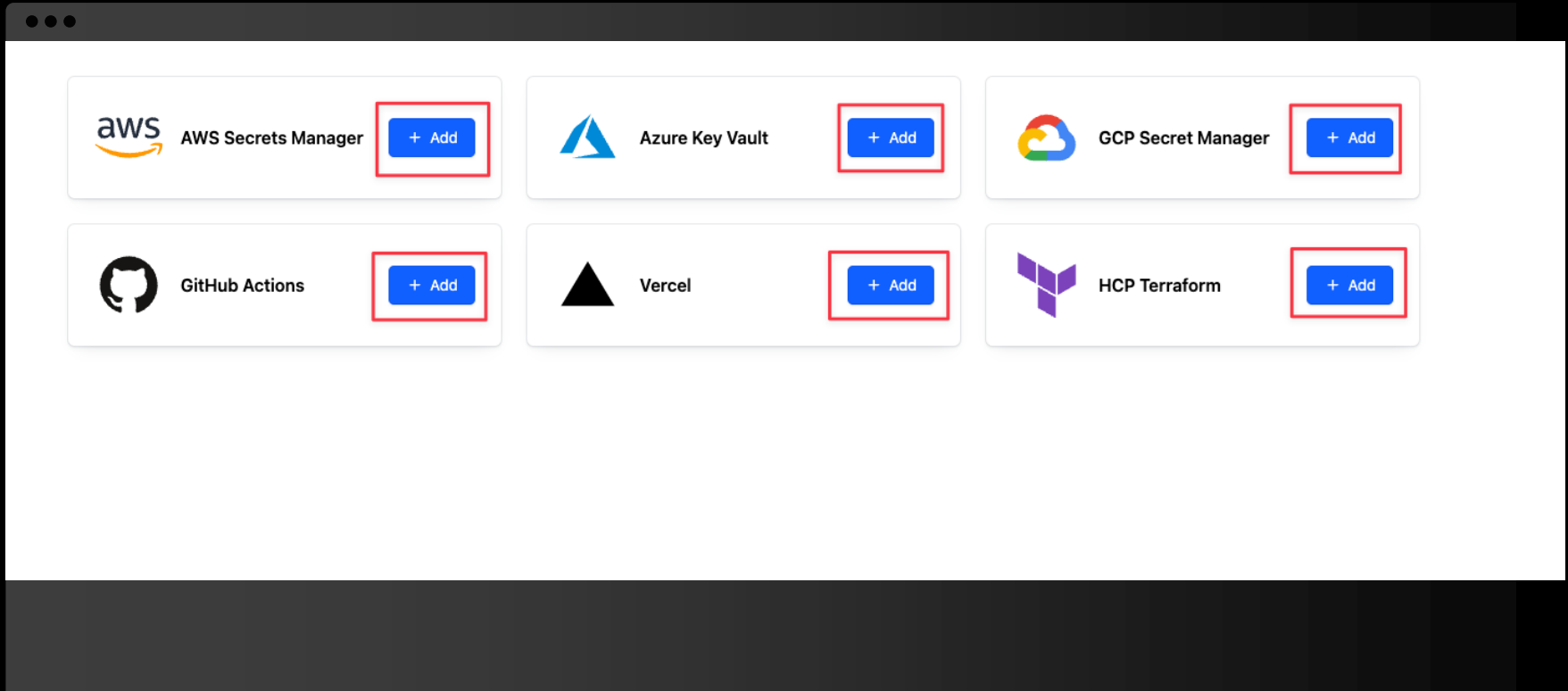
# Example UI: Add integrations



# Where this breaks for accessibility



# Each link needs a unique accessible name



# What might this code look like?

```
{{#each this.potentialIntegrations as |Integration|}}  
  <Card::Container>  
    <Icon @name={{Integration.logo}} aria-hidden="true" />  
    <div>{{Integration.name}}</div>  
    <a @route={{Integration.route}}>  
      <span aria-hidden="true">+</span>  
      <span>Add</span>  
    </a>  
  </Card::Container>  
{{/each}}
```



GitHub Actions

+ Add

# Equally viable options to consider

## The *aria-label* attribute

---

- Takes a string value
- Good option for when there is no existing element to reference
- Sometimes a bit quicker to implement
- Not additive; will replace content with its value.

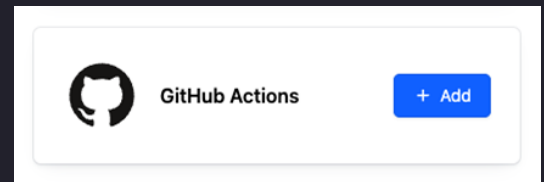
## The *aria-labelledby* attribute

---

- Takes an element's id as its value
- Can have multiple values (space separated)
- Will ignore repeated values
- Can reference...itself.
- Will replace text content with its calculated total value

# What to do instead: the *aria-label* option

```
{{!-- using aria-label --}}
{{#each this.potentialIntegrations as |Integration|}}
  <Card::Container class="integrations-card">
    <Icon @name={{Integration.logo}} aria-hidden="true" />
    <div>{{Integration.name}}</div>
    <a @route={{Integration.route}} aria-label={{concat "Add " Integration.name}}>+ Add</a>
  </Card::Container>
{{/each}}
```



# What to do instead: the *aria-labelledby* option

```
{{!-- using aria-labelledby --}}
{{#each this.potentialIntegrations as |Integration|}}
  <Card::Container>
    <Icon @name={{Integration.logo}} aria-hidden="true" />
    <div id={{Integration.nameId}}>{{Integration.name}}</div>
    <a
      @route={{Integration.route}}
      aria-labelledby={{concat Integration.actionId Integration.nameId}}
    >
      <span>+</span><span id={{Integration.actionId}}>Add</span>
    </a>
  </Card::Container>
{{/each}}
```



GitHub Actions

+ Add



# Users can navigate by element groups

Which one would **you** prefer?

**Links**

Add

Add

Add

Add

Add

**Links**

Add AWS Secrets Manager

Add Azure Key Vault

Add GitHub Actions

Add HCP Terraform

Add Vercel

# Bonus Example: Accordion using *aria-labelledby*

```
<div class="accordion-item_header">
  <button
    class={{this.classNames}}
    type="button"
    {{on "click" this.onClick}}
    aria-controls={{@contentId}}
    aria-expanded={{if @isOpen "true" "false"}}
    aria-labelledby="accordion-title"
  >
    <Hds::Icon @name="chevron-down" @size={{if (eq @size "large") "24" "16"}} />
  </button>
  <h2 id="accordion-title">{{@accordionItemTitle}}</h2>
</div>
```

# DETOUR



# The Accessible Name and Description Computation

# Detour Agenda

Context

---

AccName: Browser Instructions

---

Browser Exceptions

---

AccName: Dev Guidance

---

AccDesc: Browser Instructions

---

AccDesc: Dev Guidance

---

# 3 things to know about this spec

1. *The Accessible Name and Description Computation* is a specification that tells browsers how to calculate the accessible name and the accessible description.
2. The goal of the browser is to expose some accessible name (AccName), and in some cases, an accessible description (AccDesc) to the accessibility object.
3. They will follow specification as much as they can but they will also do weird stuff non-spec things if they are convinced they need to do to that.

# AccName: Browser Instructions

How the **browser** should find an accessible name

Browsers are supposed to return the first thing they find, looking in this order:

1. Look for an *aria-labelledby* attribute
2. Then look for an *aria-label* attribute with a non-empty value
3. Then look for an *aria-describedby* attribute
4. Look for semantic content (text of a button element. Label. Legend.)
5. Finally, if there's nothing else, look for *title* or *placeholder*.

# AccName: Browser Exceptions

Some gotchas to be aware of so you don't accidentally get things wrong

1. They must return the first thing they find....even if that thing has an "empty" string.
2. They may choose to expose text even if you've used the *aria-hidden* attribute.
3. There is no spec for how browsers should implement exceptions. This means they are not likely to be implemented consistently.



# AccName: Developer Guidance

How you, an informed **developer**, prioritizes

1. Semantic content
2. If a suitable label otherwise already exists, or you know there will be multiple values: use the *aria-labelledby*
3. If no suitable label exists: use the *aria-label* attribute
4. Don't use the *placeholder* or *title* attributes for an accessible name
5. For JS devs specifically: i18n correctly, okay?

# AccDesc: Browser Instructions

How the **browser** should find an accessible description

- Look for an *aria-describedby* attribute (if not used for AccName)
- Then look for an *aria-description* attribute
- Then look for elements that are eligible for the description calculation, and **not already used** for the accessible name computation:
  - table caption text content
  - summary element, text equivalent computation of subtree
- The *title* attribute value (again, if not already used)

# AccDesc: Developer Guidance

How you, an informed **developer**, prioritizes

1. Semantic content (i.e., Table *caption* element content)
2. Use the *aria-describedby* attribute for help and error text
  - Takes an id as the value
  - It can have multiple values (space separated id attributes)
3. The *aria-description* attribute is also a possibility
  - It should have a string value
  - It is preferred to put the descriptive text in the DOM

**END  
DETOUR**

02

# Headings, Labels, and Instructions

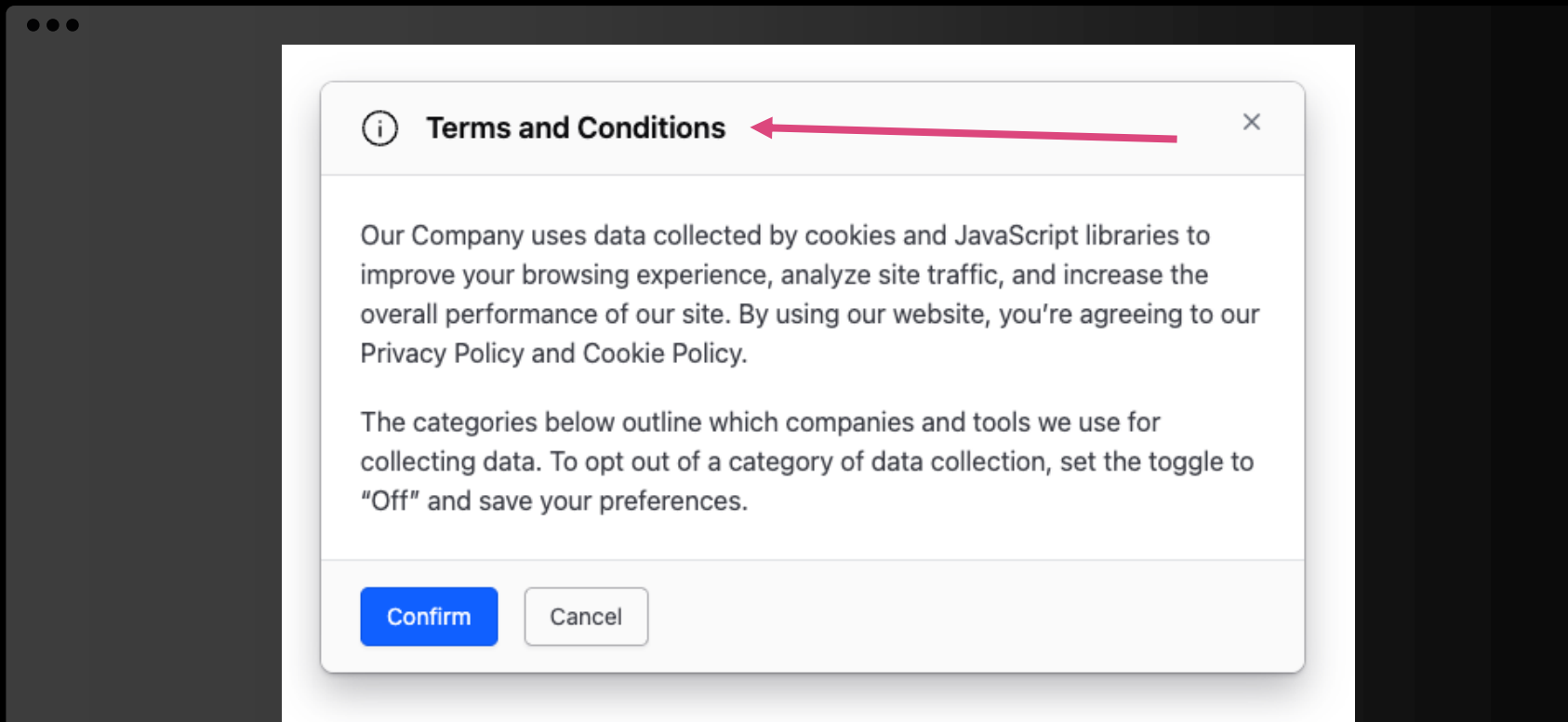
02.1

# Headings, Labels, and Instructions

# Headings TL;DR

1. HTML heading elements for semantic meaning, CSS for styling
2. Every section should have a heading
3. Headings need to be in order

# If it looks like a heading, and acts like a heading...



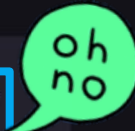


# ...but doesn't sound like a heading...

```
<use href="#flight-info-24"></use>
</svg>
<div id="ember110" class="mrs-dialog-primitive title mrs-modal title">
  <div id="ember114" class="mrs-text mrs-typography-display-300 mrs-font-weight-semibold">
    Terms and Conditions
  </div>
</div>
<button class="mrs-dismiss-button mrs-dialog-primitive__dismiss mrs-modal__dismiss" aria-label="Dismiss" type="button">
  <svg class="mrs-icon mrs-icon-x" aria-hidden="true" data-test-icon="x" fill="currentColor" id="icon-ember115"
  16" xmlns="http://www.w3.org/2000/svg">
    <use href="#flight-x-16"></use>
  </svg>
</button>
</div>
</div>
<div class="mrs-dialog-primitive__wrapper-body">
  <div class="mrs-dialog-primitive__body mrs-modal__body" tabindex="0">
    <p class="mrs-typography-body-300 mrs-foreground-primary">Our Company uses data collected by cookies and
    JavaScript libraries to improve your browsing experience, analyze site traffic, and increase the overall
    performance of our site. By using our website, you're agreeing to our Privacy Policy and Cookie Policy.</p>
  </div>
</div>
```

# ...then it's not a heading, it's an #a11yFail.

```
<use href="#flight-info-24"></use>
</svg>
<div id="ember110" class="mrs-dialog-primitive title mrs-modal title">
  <div id="ember114" class="mrs-text mrs-typography-display-300 mrs-font-weight-semibold">
    Terms and Conditions
  </div>
</div>
<button class="mrs-dismiss-button mrs-dialog-primitive__dismiss mrs-modal__dismiss" aria-label="Dismiss" type="button">
  <svg class="mrs-icon mrs-icon-x" aria-hidden="true" data-test-icon="x" fill="currentColor" id="icon-ember115"
    16" xmlns="http://www.w3.org/2000/svg">
    <use href="#flight-x-16"></use>
  </svg>
</button>
</div>
</div>
<div class="mrs-dialog-primitive__wrapper-body">
  <div class="mrs-dialog-primitive__body mrs-modal__body" tabindex="0">
    <p class="mrs-typography-body-300 mrs-foreground-primary">Our Company uses data collected by cookies and
```



<https://www.w3.org/WAI/WCAG22/Techniques/failures/F2>



# Why headings matter

- The heading tells the user what that section of the page is about
- Users with screen readers can navigate to the **next heading** with a keypress
- Users with screen readers can also navigate through a list of **all headings**
- This allows users to quickly choose what part of the page they want

Headings
1: All WCAG 2.2 Understanding Docs
2: Perceivable
3: 1.1 Text Alternatives
3: 1.2 Time-based Media
3: 1.3 Adaptable
3: 1.4 Distinguishable
2: Operable
3: 2.1 Keyboard Accessible
3: 2.2 Enough Time
3: 2.3 Seizures and Physical Reactions
3: 2.4 Navigable
3: 2.5 Input Modalities
2: Understandable
3: 3.1 Readable
3: 3.2 Predictable
3: 3.3 Input Assistance
2: Robust
3: 4.1 Compatible
2: Other Understanding documents

# Headings: What to do instead

```
fill="currentColor" cx="100" cy="100" width="24" height="24" viewBox="0 0 24 24" xmlns="http://www.w3.org/2000/svg">  
  <use href="#flight-info-24"></use>  
</svg>  
<div id="ember110" class="mrs-dialog-primitive__title mrs-modal__title">  
  <h1 id="ember114" class="mrs-text mrs-typography-display-300 mrs-font-weight-semibold">  
    Terms and Conditions  
  </h1>  
</div>  
<button class="mrs-dismiss-button mrs-dialog-primitive__dismiss mrs-modal__dismiss" aria-label="Dismiss" type="button">  
  <svg class="mrs-icon mrs-icon-x" aria-hidden="true" data-test-icon="x" fill="currentColor" height="16" width="16" viewBox="0 0 16 16" xmlns="http://www.w3.org/2000/svg">  
    <use href="#flight-x-16"></use>  
  </svg>  
</button>  
</div>  
</div>
```

02.2

# Headings, **Labels**, and Instructions

# Label requirements

Make sure you don't miss anything!

## Exist

An input label must exist.

---

WCAG Success Criteria:

3.3.2 Labels or instructions

## Correct

The markup for the label is valid.

---

WCAG Success Criteria:

1.3.1 Info & Relationships  
4.1.2 Name, Role, Value

## Clear

The label is informative.

---

WCAG Success Criteria:

2.4.6 Headings and Labels

## Visible

The visible label is in the accName.

---

WCAG Success Criteria:

2.5.3 Label in name

# Label example #1

Application Name

```
<div class="form-group">
  <label class="label">Application Name</label>
  <input
    class="input-text"
    name="application-name"
    type="text"
  />
</div>
```



WCAG 3.3.2 Labels or Instructions



WCAG 1.3.1 Info and Relationships



WCAG 4.1.2 Name, Role, Value



WCAG 2.4.6 Headings and Labels



WCAG 2.5.3 Label in Name

# Label example #2

Name

```
<div class="form-group">
  <label class="label" for="guid-001">Name</label>
  <input
    class="input-text"
    id="guid-001"
    name="input-name"
    type="text"
  />
</div>
```



WCAG 3.3.2 Labels or Instructions



WCAG 1.3.1 Info and Relationships



WCAG 4.1.2 Name, Role, Value



WCAG 2.4.6 Headings and Labels



WCAG 2.5.3 Label in Name



# Labels: what to do instead

Application Name

```
<div class="form-group">
  <label class="label" for="guid-001">
    Application Name
  </label>
  <input
    class="input-text"
    id="guid-001"
    name="application-name"
    type="text"
  />
</div>
```



WCAG 3.3.2 Labels or Instructions



WCAG 1.3.1 Info and Relationships



WCAG 4.1.2 Name, Role, Value



WCAG 2.4.6 Headings and Labels



WCAG 2.5.3 Label in Name

# Bonus: label example

Cat's Name

```
<div class="form-group">
  <label class="label" for="guid-002">Cat's Name</label>
  <input
    aria-label="What do you call your cat?"
    class="input-text"
    id="guid-002"
    name="input-cat-name"
    type="text"
  />
</div>
```

× What do you call your cat?, edit text



WCAG 3.3.2 Labels or Instructions



WCAG 1.3.1 Info and Relationships



WCAG 4.1.2 Name, Role, Value



WCAG 2.4.6 Headings and Labels



WCAG 2.5.3 Label in Name

# Bonus: let's fix it

Cat's Name

```
<div class="form-group"></div>
<label class="label" for="guid-002">Cat's Name</label>
<input
  aria-description="What do you call your cat?"
  class="input-text"
  id="guid-002"
  name="input-cat-name"
  type="text"
/>
</div>
```



WCAG 3.3.2 Labels or Instructions



WCAG 1.3.1 Info and Relationships



WCAG 4.1.2 Name, Role, Value



WCAG 2.4.6 Headings and Labels

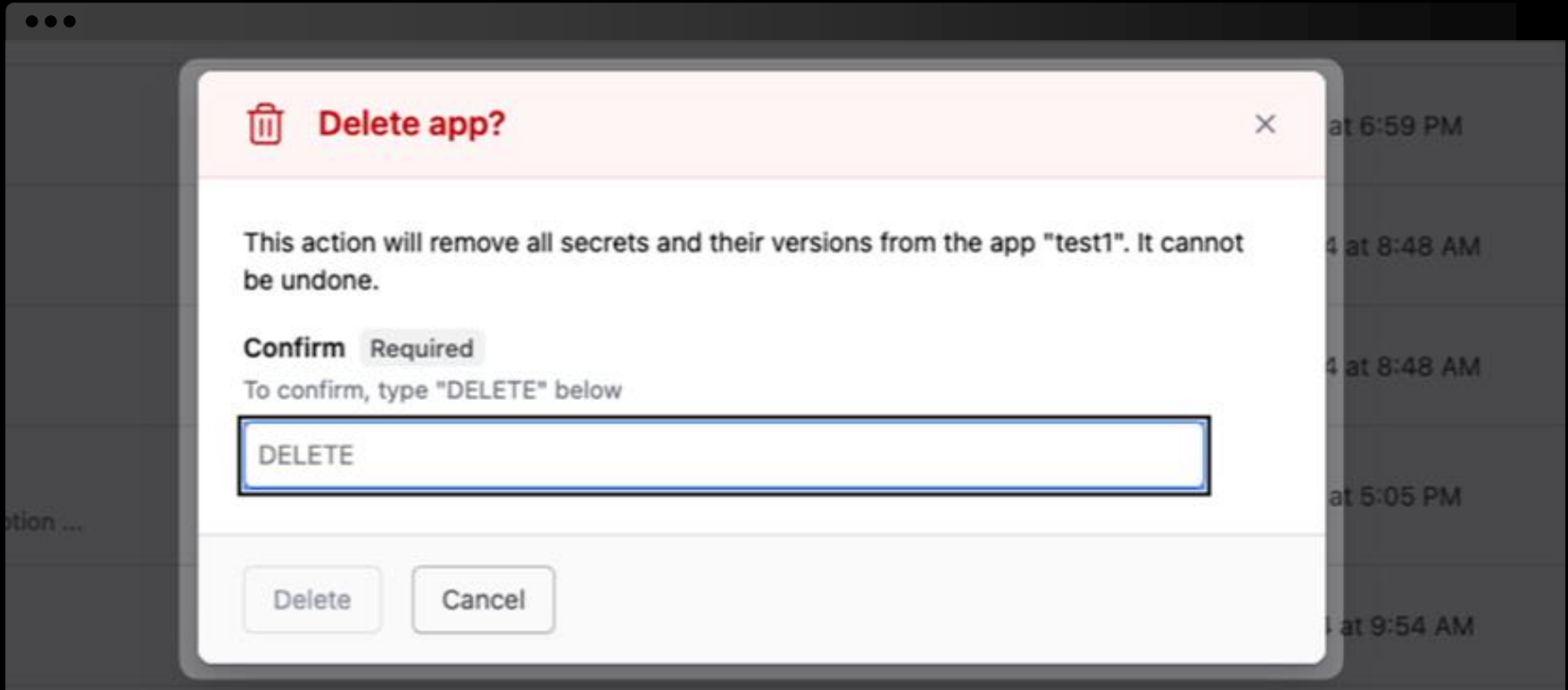


WCAG 2.5.3 Label in Name

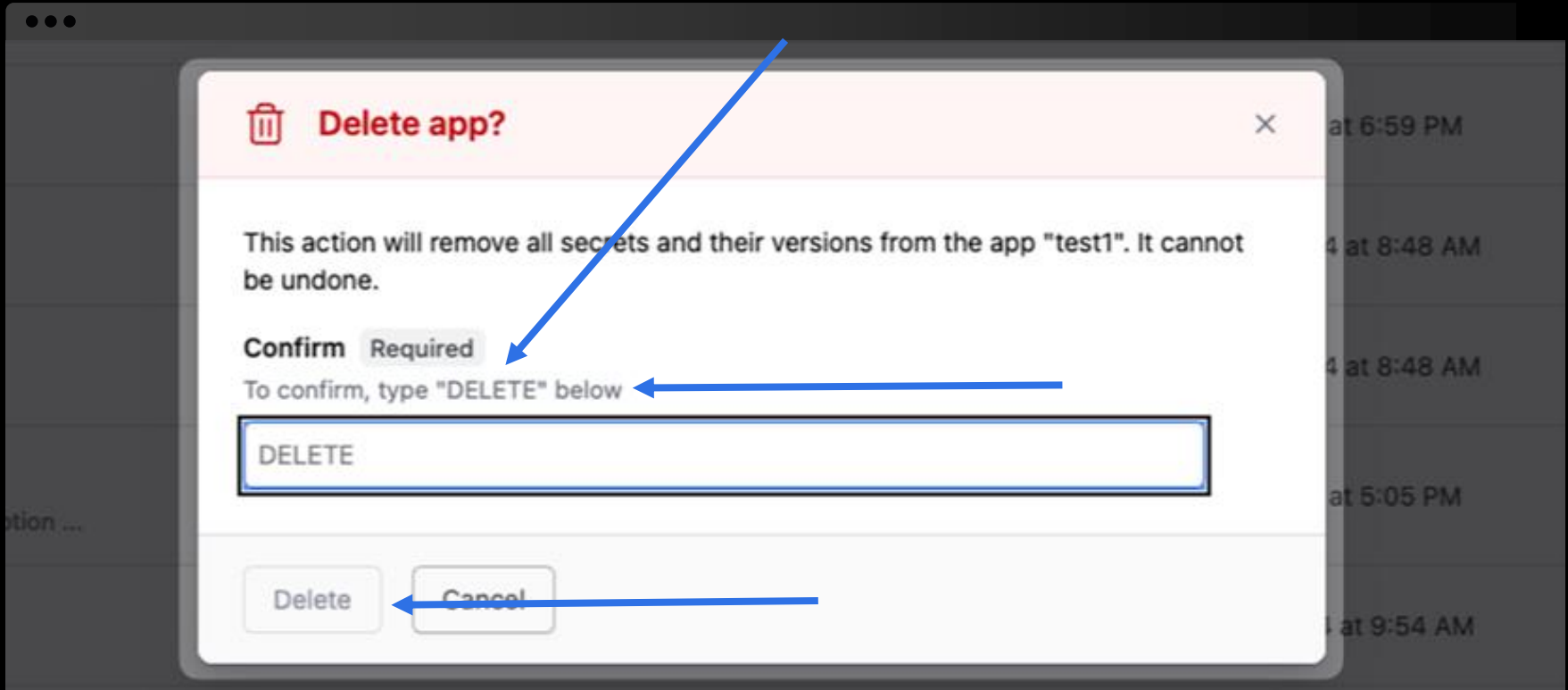
02.3

# Headings, Labels, and Instructions

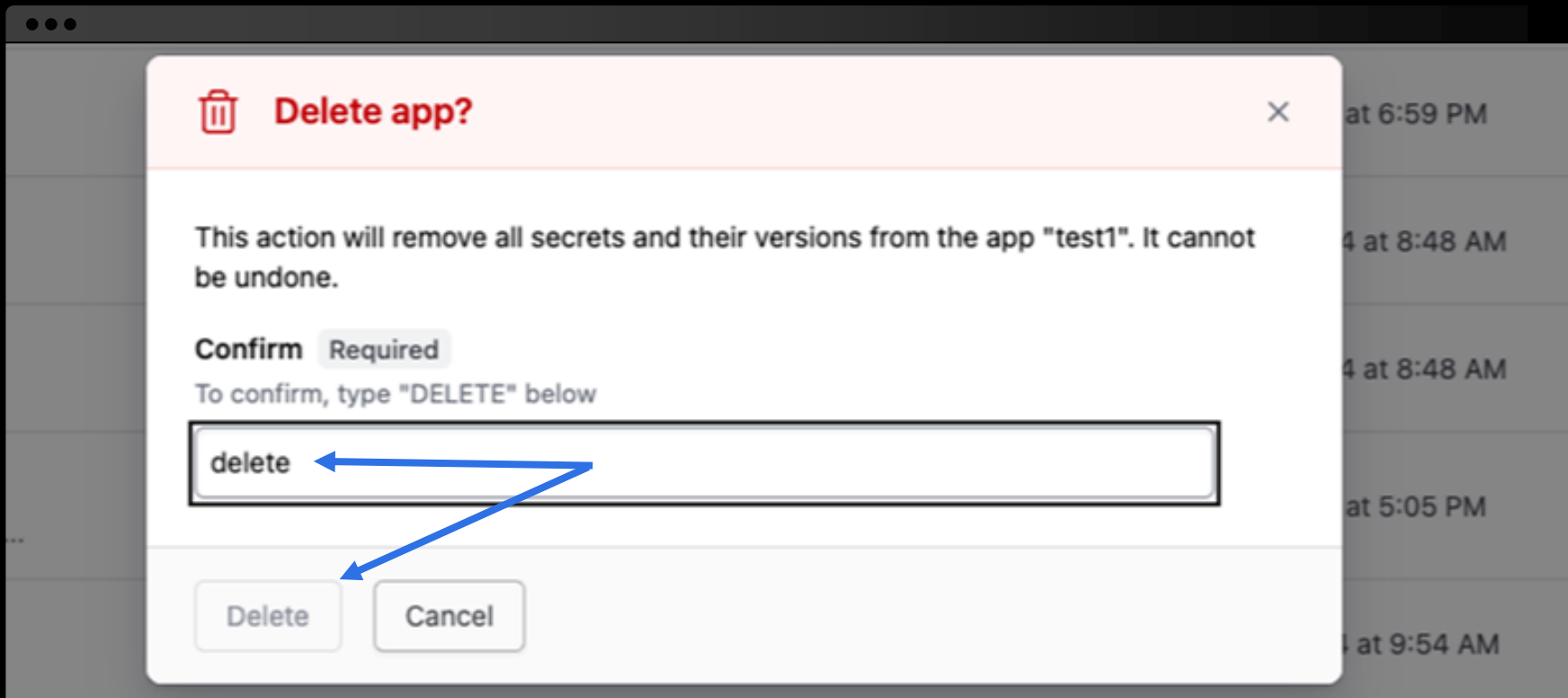
# Instructions example: delete confirmation modal



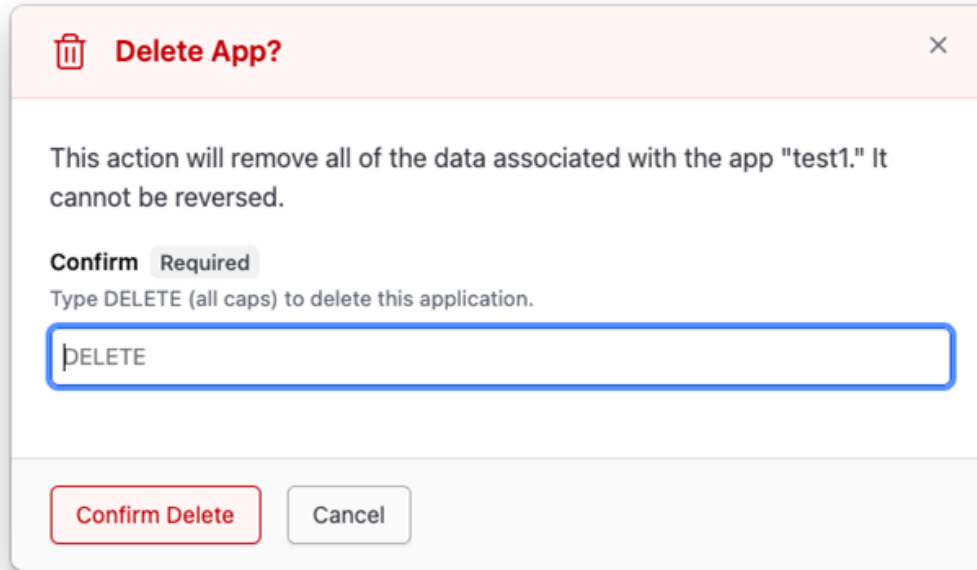
# What's wrong with this picture?




# A note about uppercase



# What to do instead: improved instructions



 **Delete App?** ×

This action will remove all of the data associated with the app "test1." It cannot be reversed.

**Confirm** Required

Type DELETE (all caps) to delete this application.

Confirm Delete Cancel



03

# Reflow, Resize, Space

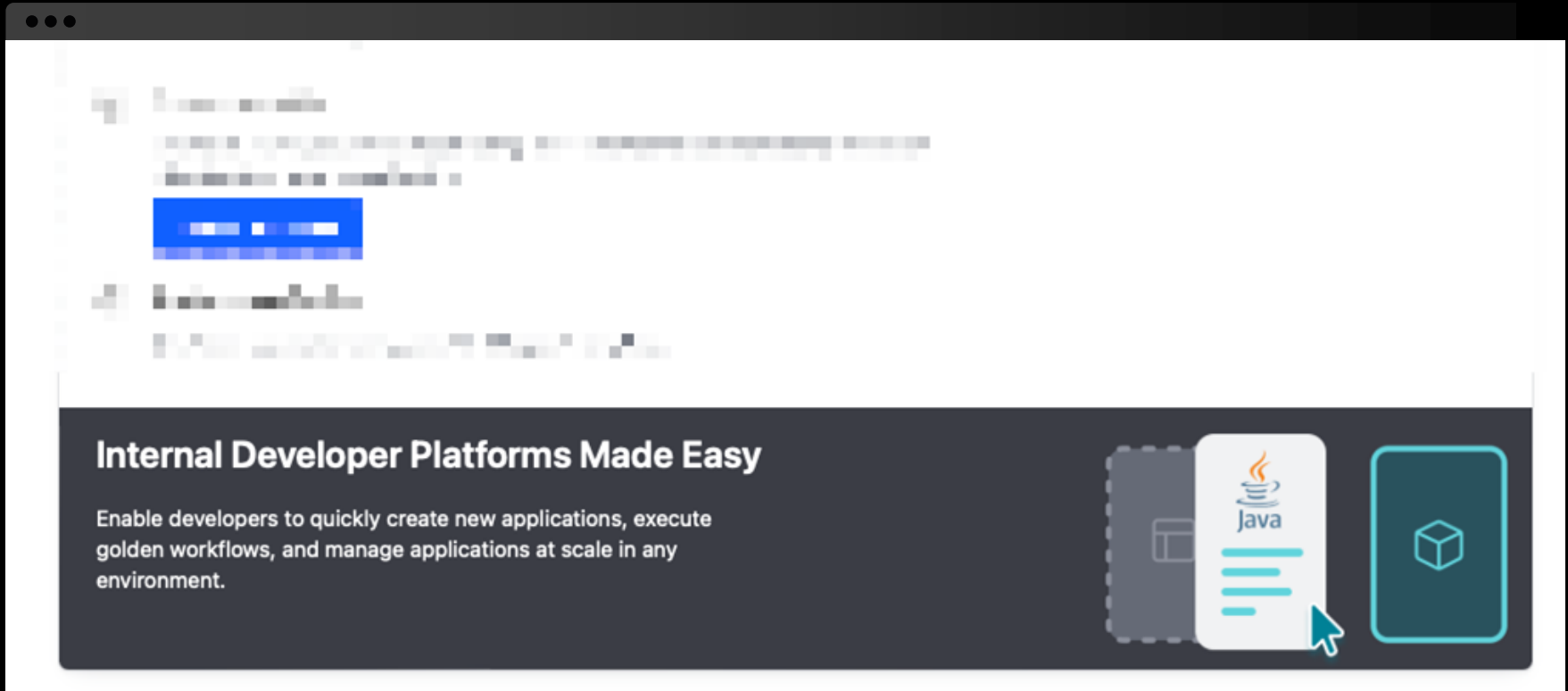
"We don't support mobile.  
We're just a desktop app."

# What we're trying to do

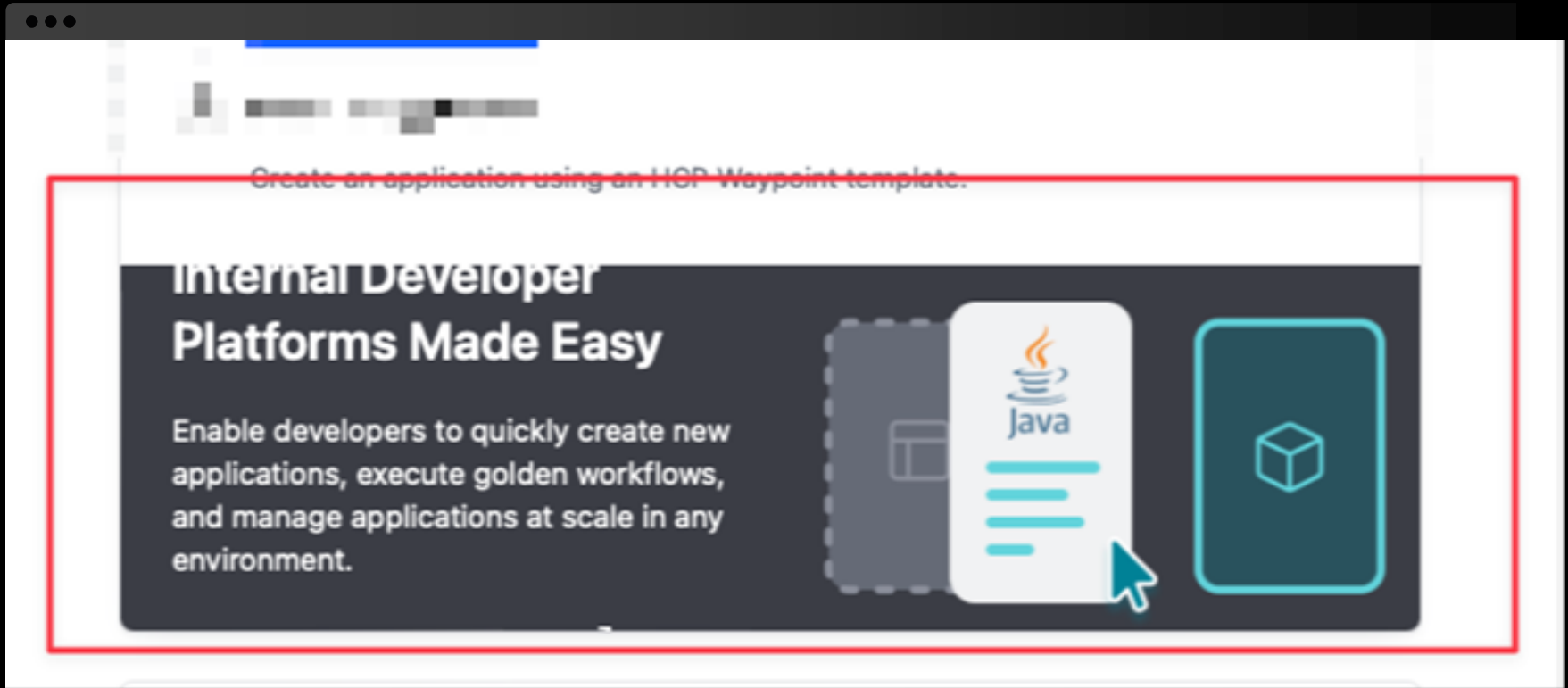
What are our **desired outcomes** for the user?

- Support browser zoom up to 400%
- Support user actions to double the text size
- Support increased space for all text

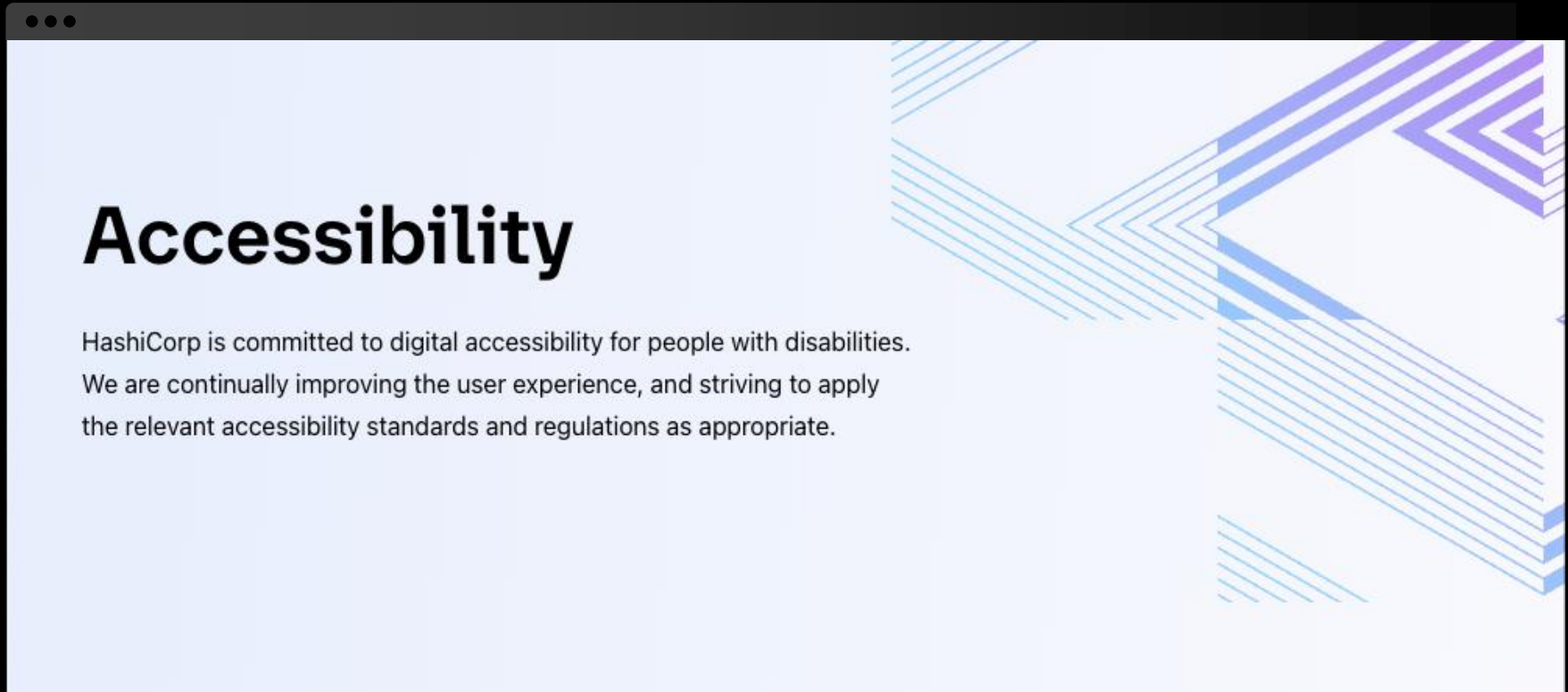
# Incorrect Reflow example (before zoom)



# Incorrect Reflow example (after zoom)



# Correct Reflow example (before zoom)



## Correct Reflow example (after zoom)

# Accessibility

HashiCorp is committed to digital accessibility for people with disabilities. We are continually improving the user experience, and striving to apply the relevant accessibility standards and regulations as appropriate.

# Text Spacing example (before)

At HashiCorp, the accessibility of our products is important to us, and aligns with our company principles. We aim for conformance with [WCAG 2.1 level AA](#), which aligns to the following standards:

- The [Web Content Accessibility Guidelines](#) (WCAG)
- [Revised 508 Standards](#)
- [EN 301 549](#)

The Voluntary Product Accessibility Template (VPAT) is a standardized form to document a product's conformance with accessibility standards. The VPAT is used for creation of Accessibility Conformance Reports (ACRs) describing the accessibility of a product's features. We use the international edition of the VPAT template, which includes the standards with which our conformance goals are aligned.



# Text Spacing example (after)

At HashiCorp, the accessibility of our products is important to us, and aligns with our company principles. We aim for conformance with [WCAG 2.1 level AA](#), which aligns to the following standards:

- The [Web Content Accessibility Guidelines](#) (WCAG)
- [Revised 508 Standards](#)
- [EN 301 549](#)

The Voluntary Product Accessibility Template (VPAT) is a standardized form to document a product's conformance with accessibility standards. The VPAT is used for creation of Accessibility Conformance Reports (ACRs) describing the accessibility of a product's features. We use the international edition of the VPAT template, which includes the standards with which our conformance goals are aligned.

# Spacing text

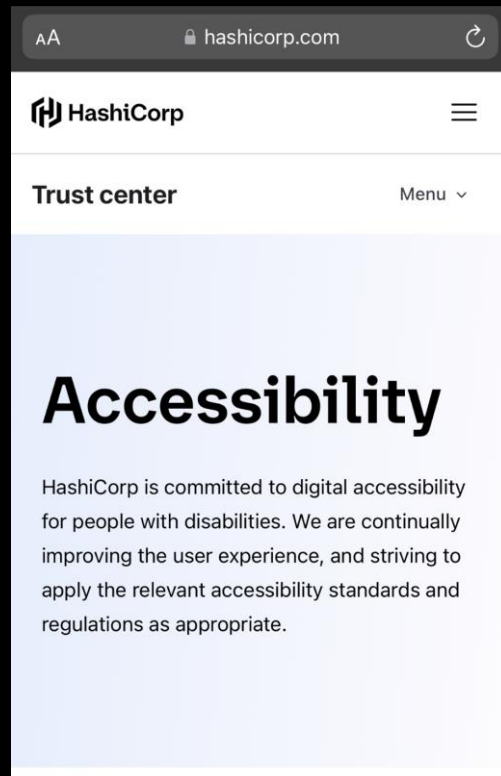
Hint: bookmarklets help here!!

1. **Line height** (line spacing) to at least 1.5 times the font size;
2. Spacing following **paragraphs** to at least 2 times the font size;
3. **Letter** spacing (tracking) to at least 0.12 times the font size;
4. **Word** spacing to at least 0.16 times the font size.

# What to do instead

Reflow, resizing, and spacing

- Use relational units (rem, em, vw, etc.)
- Use unitless line heights
- Use grid and flexbox CSS
- Test, test, test!



# Takeaway

Reflow, text resize, text spacing

If you meet these accessibility criteria,  
you sorta get some responsive design for free.\*

\*@beep: "Honestly this is how I'd frame it, so I vote *hell yeah*."

04

# Putting it all together

WCAG === Bare minimum

# What are we really trying to do here?

Think **beyond** WCAG

- Think about user outcomes
- Think about going beyond the requirements
- Consider the next level: internal requirements

# Internal Requirements

Document the places where your apps can level up

- Outline WCAG (and give links)
- Explain how your products commit to doing more
  - Design
  - Code
  - Testing
  - Release Process



Be a little selfish.

# As front-end developers...



## Try it with your keyboard

---

We are more likely to experience RSI or other movement limiting injuries. Make sure we're taking care of our future selves by ensuring keyboard navigation is supported.



## Check the color-contrast

---

Staring at a screen all day will eventually hurt your eyes. As we age, we will be less able to see color contrast. Help your future self out by checking color contrast for all users.



## Use browser zoom and see what happens

---

Again with our eyes, I know! But it happens. Check out your UIs using browser zoom...up to 400% and make sure none of the content is clipped.

*You do not require permission  
to create accessible code.*

