Web Accessibility Training



Jon Whiting & Jared Smith

WebAIM.org

Resources online at:

webaim.org/presentations/2024/january

Outline

Day 1

- Overview of WCAG 2
- Brief overview of law
- WAVE
- User experience
  - Blind (& other screen reader users)
  - Low vision
  - Colorblind

Day 2

- User experience, contd.
  - Deaf & hard of hearing
  - Photosensitive epilepsy
  - Cognitive and learning disabilities
  - Motor disabilities
- Main track: Advanced principles
  - Rules of ARIA use
  - Advanced forms
  - JavaScript
  - ARIA applied
- *Breakout option: Word, PowerPoint, and Acrobat Pro*

Is your web content accessible?

WCAG 2

- Web Content Accessibility Guidelines
- Versions 1.0 (1999), 2.0 (2008), and 2.1 (2018)
- WCAG 2.2 (2023)
- Principles based
  - **P**erceivable
  - **O**perable
  - **U**nderstandable
  - **R**obust

WebAIM WCAG 2 Checklist

Operable

Interface forms, controls, and navigation are operable.

Guideline 2.1

Make all functionality available from a keyboard.

2.1.1 Keyboard

A 2.0

- ☐ All page functionality is available using the keyboard, unless the functionality cannot be accomplished in any known way using a keyboard (e.g., free hand drawing).
- ☐ Page-specified shortcut keys and accesskeys (accesskey should typically be avoided) do not conflict with existing browser and screen reader shortcuts.

2.1.2 No Keyboard Trap

A 2.0

- ☐ Keyboard focus is never locked or trapped at one particular page element. The user can navigate to and from all navigable page elements using only a keyboard.

2.1.3 Keyboard (No Exception)

AAA 2.0

- ☐ All page functionality is available using the keyboard.

2.1.4 Character Key Shortcuts

A 2.1

- ☐ If a keyboard shortcut uses printable character keys, then the user must be able to disable the key command, change the defined key to a non-printable

Legal requirements

- Section 508
  - US federal agencies
  - WCAG 2.0
- Americans with Disabilities Act
  - 3 relevant sections – employment, state/local government, public & commercial facilities
  - No technical accessibility standard
  - Complaints reference WCAG 2.0 or 2.1
- EN 301 549
  - EU public sector
  - WCAG 2.1
- Accessible Canada Act
  - Federal Government
  - WCAG 2.1
- Other laws reference (or do not exceed) WCAG 2

User experience

Screen reader users

- Not all screen reader users are completely blind
  - Most have low vision
  - Some have a reading or cognitive disability
- Focus on structure and semantics

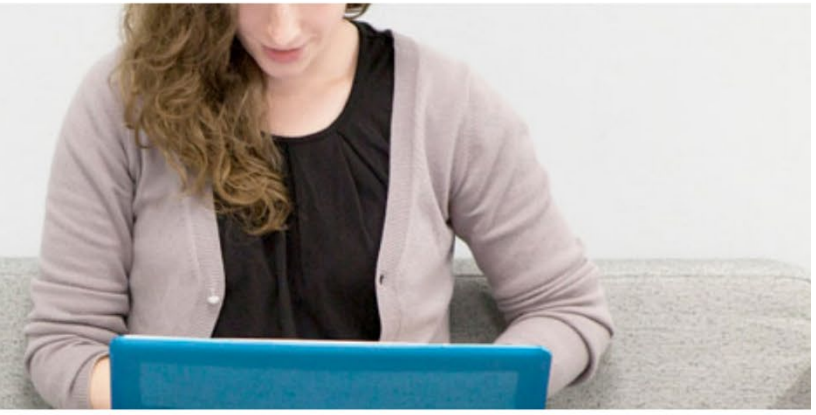
WebAIM
web accessibility in mind

services articles resources projects community

Search:

Introduction to Web Accessibility

WebAIM Training



We have web accessibility in mind

Expanding the potential of the web for people with disabilities by empowering individuals and organizations to create accessible content.

Accessibility Training

Whether here in Utah or on-site at your organization, WebAIM can provide web and document training to fit your needs.

Accessible Site Certification

As a respected third party accessibility expert, WebAIM can evaluate and certify your site to established web accessibility guidelines.

StrategicA11y

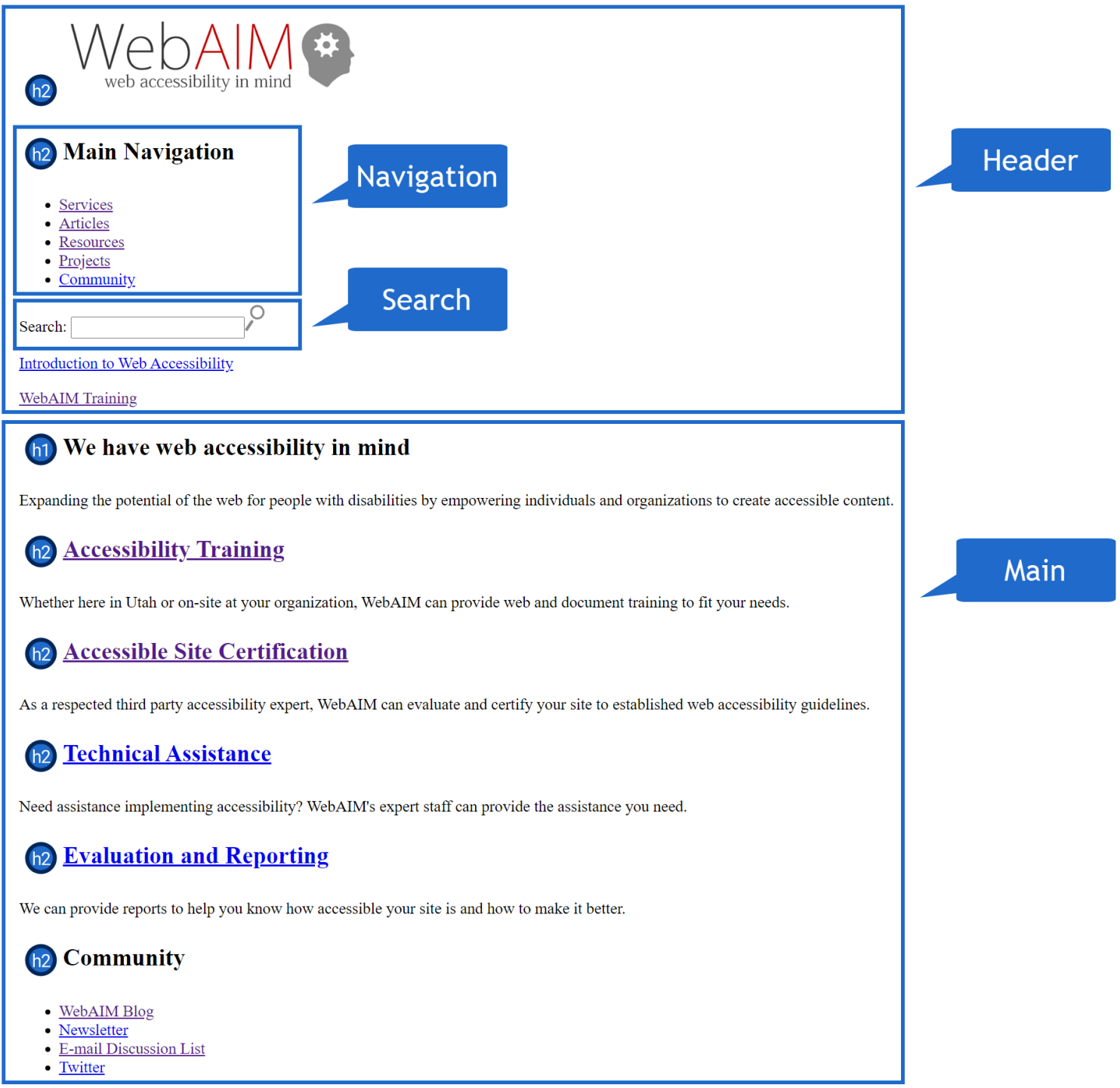
The Strategic Web Accessibility Workshop helps participants optimize their organization's accessibility.

Evaluation and Reporting

We can provide reports to help you know how accessible your site is and how to make it better.

Community

- WebAIM Blog
- Newsletter
- E-mail Discussion List
- Twitter



Navigation

Search

Header

Main

Structure & Semantics

Recommended screen readers for testing

Desktop

- Windows
  - JAWS – 54%, $1000+
  - NVDA – 31%, Free
- VoiceOver for Mac – 7%

Mobile

- VoiceOver for iOS – 72%
- Talkback for Android – 25%

Use WebAIM tutorials to learn commands and practice common tasks

Screen Reader Demo

Screen reader testing

- Don’t try to be an “expert.”
  - 10 shortcuts to get started, 20 is about all you’ll ever need.
- Screen readers read things differently and can misbehave.
- Verbosity and punctuation settings impact what is read.
  - Keep default settings.
- Focus on document structure, navigation, forms, and dynamic content.

Headings

- The most common method for finding information on a page
  - 69% of WebAIM screen reader user survey respondents
- WCAG failures:
  - Text that should be a heading, but isn’t
  - Text that shouldn’t be a heading, but is
  - Doesn’t align with visual presentation E.g.,
    H2 – Main topic
      H1 – Subtopic
  - Headings that are empty or not descriptive

Headings – Best practice

- Page “outline”
- One <h1> per page (usually)
- Do not skip heading levels E.g.,
  H2 – Topic
    H4 – Subtopic

Regions and Landmarks



Introduction to Web Accessibility

Last updated: Apr 14, 2020

Translations

Russian by EduTranslator

Related Resources

- Constructing a POUR Website
- Visual
- Hearing
- Motor
- Cognitive
- Seizure
- Design Considerations

Article Contents

- Introduction
- People with Disabilities on the Web
- Implementing Web Accessibility
- Principles of Accessible Design
- Conclusion

Introduction

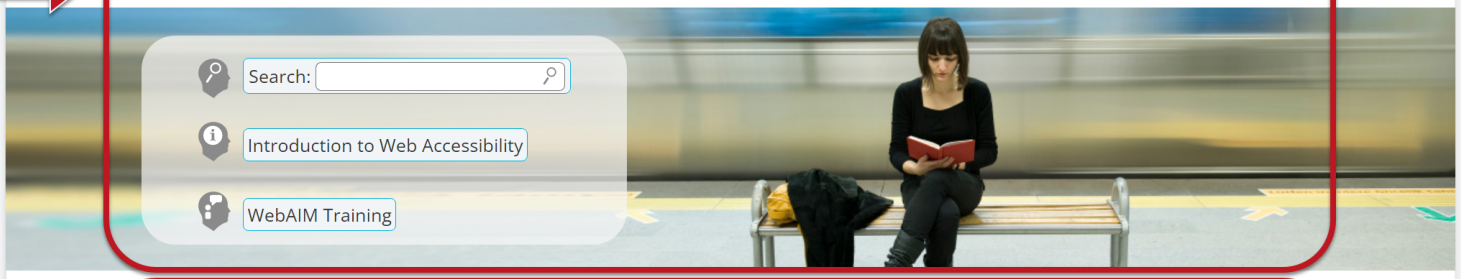
Most of us can hardly conceive of life without the internet. Some have argued that no other single invention has been more revolutionary since Gutenberg's printing press in the 1400s. The world can be "at your fingertips" at the click of a

Images in Introduction section ©2016 W3C®, Creative Commons Attribution license.

<header>

WebAIM
web accessibility in mind

services articles resources projects community



</header>

<main>

Introduction to Web Accessibility

Last updated: Apr 14, 2020

Home > Articles > Introduction to Web Accessibility

Translations

Russian by EduTranslator

Article Contents

- Introduction
- People with Disabilities on the Web
- Implementing Web Accessibility
- Principles of Accessible Design
- Conclusion

Related Resources

- Constructing a POUR Website
- Visual
- Hearing
- Motor
- Cognitive
- Seizure
- Design Considerations

Introduction

Most of us can hardly conceive of life without the internet. Some have argued that no other single invention has been more revolutionary since Gutenberg's printing press in the 1400s. The world can be "at your fingertips" at the click of a

Images in Introduction section ©2016 W3C®, Creative Commons Attribution license.

</main>

<footer>

©2023 WebAIM

Institute for Disability Research, Policy, and Practice
Utah State University
6807 Old Main Hill
Logan, UT 84322-6807
435.797.7024

From the Blog

- Captioning and Sign Language Interpretation in Zoom: Features and Pitfalls
- Tips for Getting the Most Out of WAVE
- Flexbox and the Screen Reader Experience
- Tips for Accessible Screen Sharing

Popular Resources

- WebAIM Training
- WCAG 2 Checklist
- WebAIM Monthly Newsletter
- Color Contrast Checker
- Web Accessibility for Designers
- WAVE Web Accessibility Evaluation Tool

Check Your Accessibility

Web site address:

WAVE

Contact About RSS Feeds Twitter Copyright & Terms of Use

</footer>


<header>
WebAIM
web accessibility in mind
<nav>
services
articles
resources
projects
community
Search:
Introduction to Web Accessibility
WebAIM Training
<main>
Introduction to Web Accessibility
Home > Articles > Introduction to Web Accessibility
<nav>
Article Contents
Introduction
People with Disabilities on the Web
Implementing Web Accessibility
Principles of Accessible Design
Conclusion
<aside>
Last updated: Apr 14, 2020
Translations
Russian by EduTranslator
Related Resources
Constructing a POUR Website
Visual
Hearing
Motor
Cognitive
Seizure
Design Considerations
Introduction
Most of us can hardly conceive of life without the internet. Some have argued that no other single invention has been more revolutionary since Gutenberg's
Images in Introduction section ©2016 W3C®, Creative Commons Attribution license.
<footer>
©2023 WebAIM
Institute for Disability Research, Policy, and Practice
Utah State University
6807 Old Main Hill
Logan, UT 84322-6807
435.797.7024
From the Blog
Captioning and Sign Language Interpretation in Zoom: Features and Pitfalls
Tips for Getting the Most Out of WAVE
Flexbox and the Screen Reader Experience
Tips for Accessible Screen Sharing
Popular Resources
WebAIM Training
WCAG 2 Checklist
WebAIM Monthly Newsletter
Color Contrast Checker
Web Accessibility for Designers
WAVE Web Accessibility Evaluation Tool
Check Your Accessibility
Web site address:
WAVE
Contact
About
RSS Feeds
Twitter
Copyright & Terms of Use

ARIA
Accessible Rich Internet Applications

ARIA expands the vocabulary of HTML

Rule #1 of ARIA Use

“If you can use a native HTML element or attribute…then do so.”

Using ARIA (W3C)

Rule #1, paraphrased

If you can use HTML, don’t use ARIA

ARIA Landmarks

- `<header>` - `role="banner"`
- `<nav>` - `role="navigation"`
- `<main>` - `role="main"`
- `<aside>` - `role="complementary"`
- `<footer>` - `role="contentinfo"`
- `<search>` - `role="search"`
  - Newly added to HTML. Not yet fully supported.
  - Don't confuse with **`<input type="search">`**.

Regions/Landmarks

- All content should be in a region
- More doesn’t always mean better
- Region/Landmark are interchangeable to screen readers
- We recommend HTML regions (Rule #1)
  - `<header role="banner">`, etc. is unnecessary
  - `<search role="search">` is a good idea, for now

role="search"

WebAIM
web accessibility in mind

services articles resources projects community



Introduction to Web Accessibility

Home > Articles > Introduction to Web Accessibility

Last updated: Apr 14, 2020

Translations

Russian by EduTranslator

Related Resources

- Constructing a POUR Website
- Visual
- Hearing
- Motor
- Cognitive
- Seizure
- Design Considerations

Article Contents

- Introduction
- People with Disabilities on the Web
- Implementing Web Accessibility
- Principles of Accessible Design
- Conclusion

Introduction

Most of us can hardly conceive of life without the internet. Some have argued that no other single invention has been more revolutionary since Gutenberg's printing press in the 1400s. The world can be "at your fingertips" at the click of a

Images in Introduction section ©2016 W3C®, Creative Commons Attribution license.

©2023 WebAIM

Institute for Disability Research, Policy, and Practice
Utah State University
6807 Old Main Hill
Logan, UT 84322-6807
435.797.7024

Check Your Accessibility

Web site address:

WAVE

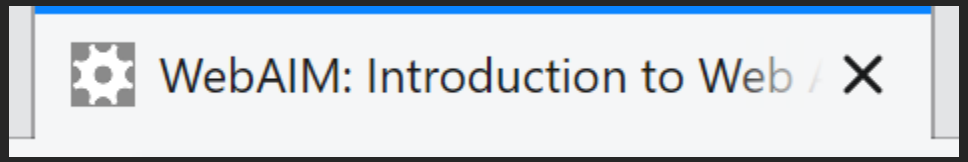
From the Blog

- Captioning and Sign Language Interpretation in Zoom: Features and Pitfalls
- Tips for Getting the Most Out of WAVE
- Flexbox and the Screen Reader Experience
- Tips for Accessible Screen Sharing

Popular Resources

- WebAIM Training
- WCAG 2 Checklist
- WebAIM Monthly Newsletter
- Color Contrast Checker
- Web Accessibility for Designers
- WAVE Web Accessibility Evaluation Tool

Page Title




```
<title>WebAIM: Introduction to
Web Accessibility</title>
```

- Often the first thing seen or read by a screen reader
- Must be descriptive
- Should be succinct
- Should usually match, or be similar to, the first-level heading

Page title article

WebAIM Web Accessibility Evaluation Guide

WAVE

+ Keyboard testing

+ Browser zoom

+ Contrast tools

+ DevTools

Etc.

Forms

- ,  - Ensure form controls have descriptive, correctly-associated labels and that `<label>` is used when possible.
  - If a form control gains focus or is activated when its label text is **clicked** with a mouse, then it is likely that the text is a correctly-associated `<label>`.
- ,  - If a label is not visible, check for a hidden label, `aria-label`, or `title` attribute.
-  – If a `<select>` menu does not have a `<label>`, confirm that the default option presents the purpose of the menu.
- , , ,  - Use the **Code panel** to explore broken labels.
- , ,  - Look for `<fieldset>`/`<legend>` where a higher-level description for a group of form fields is needed.
- Required fields are visually apparent and are presented to a screen reader through at least one of the following: label/accessible name, `aria-required`, or the `required` attribute.
- Trigger form errors and then test error feedback using the **WAVE browser extension**.
-  – Descriptions and inline error messages are associated to the correct form control using `aria-describedby`.

Tables



wave.webaim.org

Evaluate headings, regions/landmarks, and page title

WAVE “Structure” tab

Lists

- `<ol>`/Numbers for sequence or hierarchy
- `<ul>`/Bullets for parallel or equal importance
- `<dl>` for “name/value pairs”
  - E.g., FAQ or glossary

EASY SHORTBREAD



Ingredients

- 1 cup butter, softened
- 1⁄2 cup sugar
- 2 1⁄2 cups flour

Directions

1. Preheat the oven to 300 F.
2. Cream butter and sugar.
3. Gradually stir flour into the creamed mixture until blended.
4. Pat the dough in the bottom of an ungreased 9x13 baking pan.
5. Bake at 300 F for 30 to 40 minutes, until just lightly browned.
6. Remove from oven and pierce all over with a fork.

Baking Terms

Softened butter

Butter left at room temperature for approximately 20 minutes.

Cream butter and sugar

Mix butter and sugar together until it is light and fluffy.

kindle fire HD

Our most affordable Kindle Fire, now in HD





See 1 customer image

Share your own customer images

learn more 17 of 17

(9,193 customer reviews)

$139.00 Prime Exclusive digital benefits with Amazon Prime

Storage Size: 8 GB

8 GB | 16 GB

Offer Type: With Special Offers

With Special Offers | Without Special Offers

Special offers and sponsored screensavers display on the Kindle Fire lock screen. Learn more

In Stock.
Ships from and sold by **Amazon Digital Services**.
Important information about using outside the U.S.

Want it tomorrow, Feb. 19? Order within **7 hrs 50 mins** and choose **One-Day Shipping** at checkout. Details

We want you to know...

Learn more about design decisions and feature changes to help make an informed purchase

Our most affordable Kindle Fire—now with a stunning HD display, faster processor, and longer battery life

- Experience movies, TV, and games, and more on a stunning HD display (216 ppi / 1280x800)
- Fast 1.5GHz dual-core processor—apps launch quickly, games and videos play smoothly
- Create profiles and set time limits for children with Kindle FreeTime. Easy-to-use parental controls let everyone enjoy, worry-free
- Ultra-fast web browsing over built-in Wi-Fi, plus updated e-mail and calendar support for Gmail, Outlook, and more
- Instant access to over 100,000 apps and games in the Amazon Appstore, including a new paid app for free every day
- Prime In addition to Free Two-Day Shipping, Prime members can stream tens of thousands of Prime Instant Video titles at no additional cost, over half of which can be downloaded to the latest generation of Kindle Fire tablets. Learn more

Meaningful link text

- Click here
- Click here to log in
- Click here to log in
- Log in

Tables

Data tables

Class Schedule

Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220

Parts of accessible tables

1. Caption (if present)
2. Headers
3. Scope

```
<table>
<caption>Class Schedule</caption>
<tr>
  <th scope="col">Class Name</th>
  <th scope="col">Course Number</th>
  <th scope="col">Location</th>
</tr>
<tr>
  <th scope="row">Advanced Website
Development</th>
  <td>BIS 5650</td>
  <td>B105</td>
</tr>
...
```

Structure for screen readers

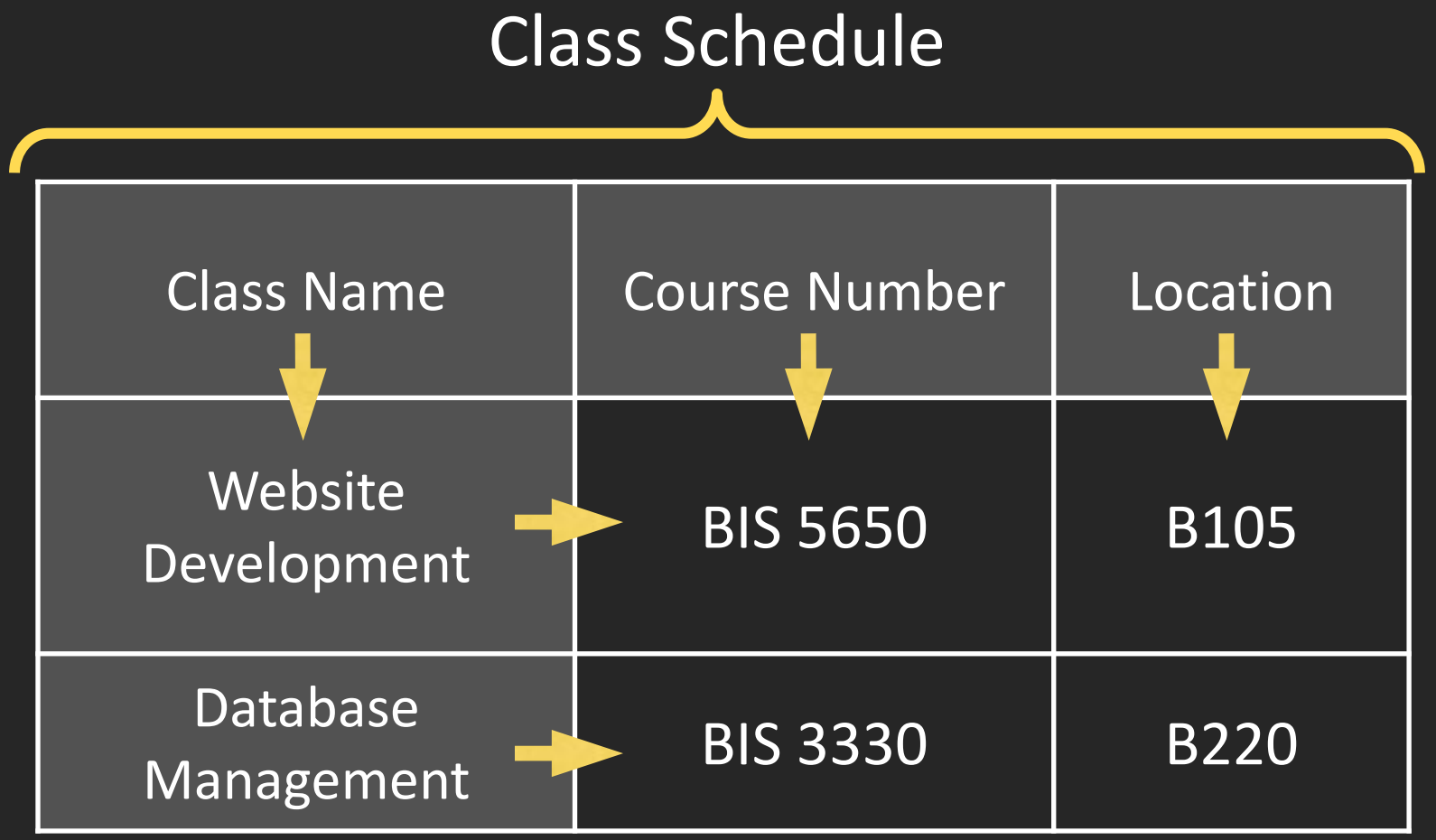


Table example

Spanned cells and multiple header levels

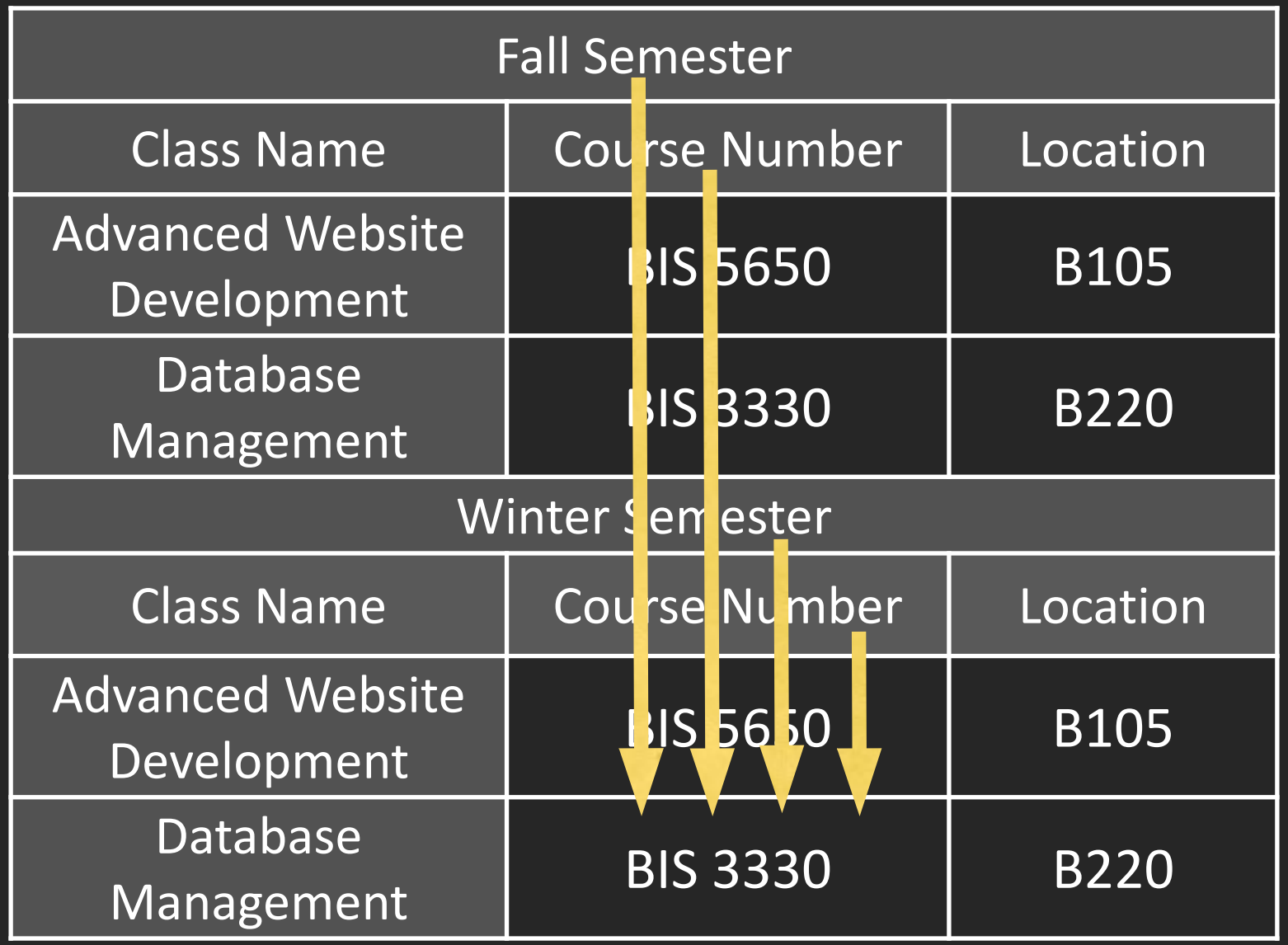
Section	Course #	Day	Start time	End time
BIO	110	M,W,F	08:00	9:00
	120	T,Th	09:00	11:00
BUS	110	T,Th	10:00	11:30
	210	M,W,F	12:00	13:00

Fall Semester		
Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220

Fall Semester

Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220

Fall Semester		
Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220
Winter Semester		
Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220


Fall Semester
Class Name
Course Number
Location
Advanced Website Development
BIS 5650
B105
Database Management
BIS 3330
B220
Winter Semester
Class Name
Course Number
Location
Advanced Website Development
BIS 5650
B105
Database Management
BIS 3330
B220

Fall Semester

Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220

Winter Semester

Class Name	Course Number	Location
Advanced Website Development	BIS 5650	B105
Database Management	BIS 3330	B220

Language

“Most people today can hardly conceive of life without the internet. Some have argued that no other single invention has been more revolutionary since Gutenberg's printing press in the 1400s. Now, at the click of a mouse, the world can be “at your fingertips”—that is, if you can use a mouse... and see the screen... and hear the audio—in other words, if you don't have a disability of any kind.”



Language of Page (Level A)

```
<html lang="en">
```

WebAIM language article

Language of Parts (Level AA)

Language of parts example

Alternative Text

Alternative text

- Read by screen readers
- Alternative to images if they are disabled or not supported
- Used by search engines

Alternative text best practice

- Must
  - Be equivalent
- Should
  - Usually be succinct
  - Use correct grammar and punctuation
- Should not
  - Be redundant
  - Start with "image of ..." or "graphic of ...“

What is **equivalent** alternative text?

CONTENT and **FUNCTION**

VERY RARELY Description

Alternative text for this image?





“WebAIM - web accessibility in mind”

Text alternatives can be presented two ways

- In the **alt attribute** of the img element.
  `<img alt="I am alt text">`
- In the **context** or surroundings of the image itself.




```
<a href="minivans.html">
      <img src="minivans.jpg" alt="???"><br>
      Minivans
</a>
```




```
<a href="minivans.html">
      <img src="minivans.jpg" alt=""><br>
      Minivans
</a>
```

Empty vs. Missing alt attribute

- Empty alt (alt="") is a statement:
  - The image is decorative, or
  - Alternative text is nearby
- Missing alt is ambiguous
  - Screen readers may or may not treat it the same as alt=""
- Content tools (e.g., WordPress and Word) should have a way to identify decorative images

Images that are the only thing within a link must **ALWAYS** have alternative text

... and image map hot-spots and image buttons too.

Image Link



Text Link

```
<a href="minivans.html">
<img src="minivans.jpg" alt="Minivans"></a>
<br>
<a href="minivans.html">Minivans</a>
```

Download the Employment Application 

```
<a href="app.pdf">Download the
Employment Application
<img src="acrobat.png" alt="???"></a>
```

Download the Employment Application 

```
<a href="app.pdf">Download the
Employment Application
<img src="acrobat.png" alt="PDF"></a>
```

The unexpected star of NASA's Webb images — the alt text descriptions

A team in Baltimore was responsible for the words that made the stunning photos accessible to everyone



Perspective by Theresa Vargas
Metro columnist

July 20, 2022 at 5:09 p.m. EDT



This image released by NASA on July 12 was captured by the James Webb Space Telescope. (AP)

Comment Gift Article Share

In the days since NASA publicly shared stunning images captured by the James Webb Space Telescope, people have oohed and aahed. They have

The unexpected star of NASA's Webb images — the alt text descriptions

A team in Baltimore was responsible for the words that made the stunning photos accessible to everyone



Perspective by Theresa Vargas
Metro columnist

July 20, 2022 at 5:09 p.m. EDT



This image released by NASA on July 12 was captured by the James Webb Space Telescope. (AP)

Comment Gift Article Share

In the days since NASA publicly shared stunning images captured by the James Webb Space Telescope, people have oohed and aahed. They have

```
alt=""
```

The unexpected star of NASA's Webb images — the alt text descriptions

A team in Baltimore was responsible for the words that made the stunning photos accessible to everyone



Perspective by Theresa Vargas
Metro columnist

July 20, 2022 at 5:09 p.m. EDT



This image released by NASA on July 12 was captured by the James Webb Space Telescope. (AP)

Comment Gift Article Share

In the days since NASA publicly shared stunning images captured by the James Webb Space Telescope, people have oohed and aahed. They have

```
alt="Carina Nebula"
```

WEBB
SPACE TELESCOPE

SEARCH MENU

FILTER IMAGES

Category | Type | Collection | Enter Keyword | SUBMIT

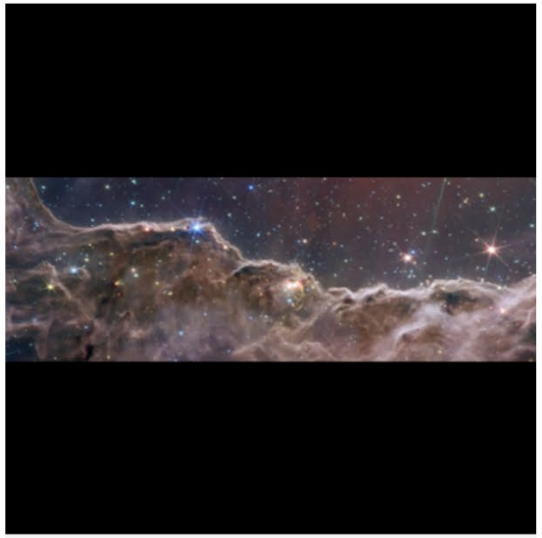
First Previous 1 2 3 4 5 Next Last



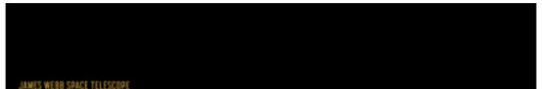
Webb's Science Themes

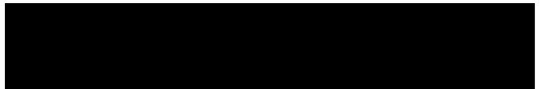


"Cosmic Cliffs" in the Carina Nebula (NIRCam Image)

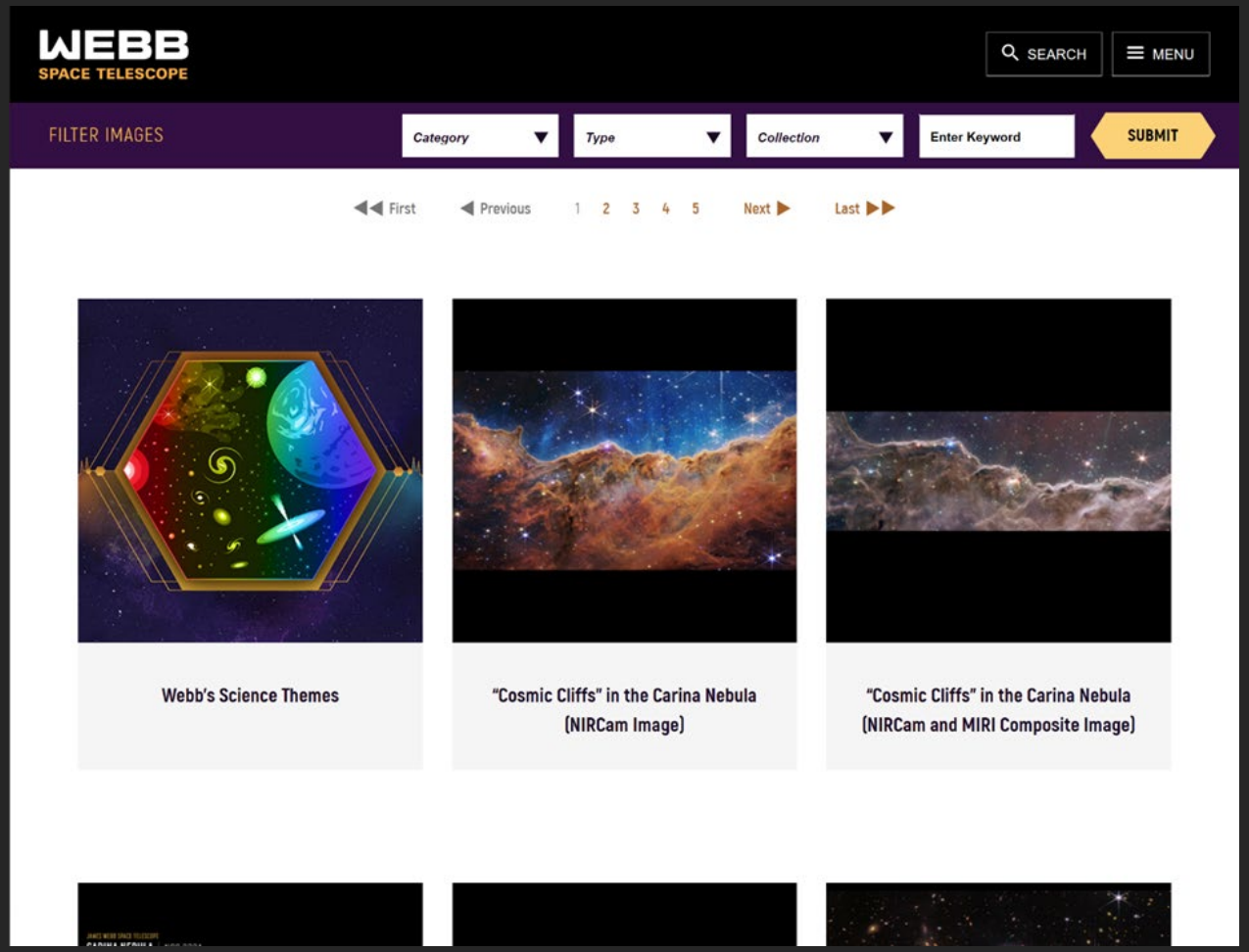


"Cosmic Cliffs" in the Carina Nebula (NIRCam and MIRI Composite Image)









alt="Image of bright eight-pointed stars scattered across a dense orange-brown undulating cloudscape below a deep blue starscape."

“Cosmic Cliffs” in the Carina Nebula (NIRCam Image)

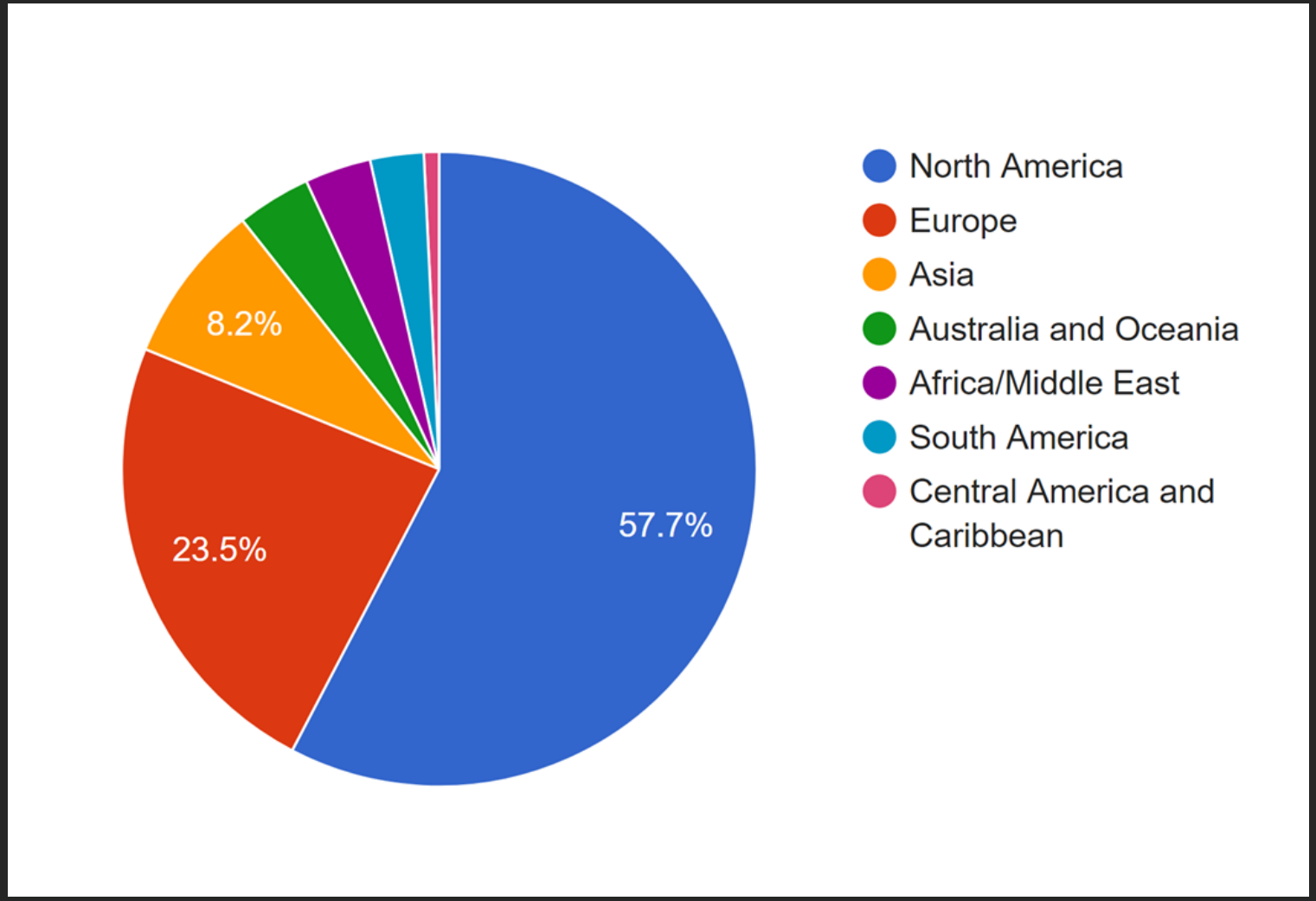
< VIEW ALL IMAGES





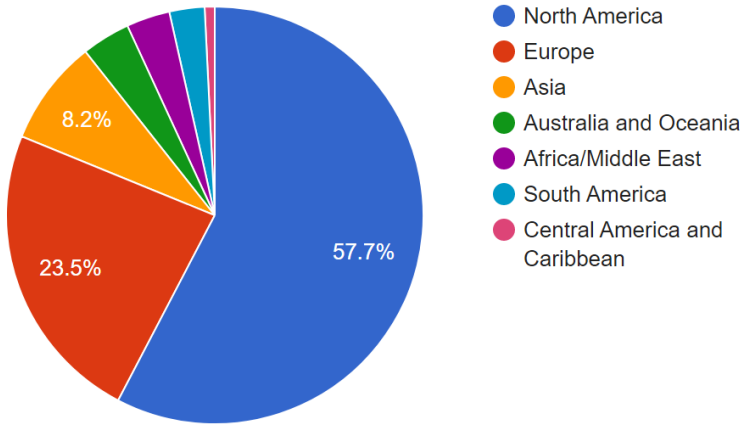
alt="The image is divided horizontally by an undulating line between a cloudscape forming a nebula along the bottom portion and a comparatively clear upper portion. Speckled across both portions is a starfield, showing innumerable stars of many sizes. The smallest of these are small, distant, and faint points of light. The largest of these appear larger, closer, brighter, and more fully resolved with 8-point diffraction spikes. The upper portion of the image is blueish, and has wispy translucent cloud-like streaks rising from the nebula below. The orangish cloudy formation in the bottom half varies in density and ranges from translucent to opaque. The stars vary in color, the majority of which, have a blue or orange hue. The cloud-like structure of the nebula contains ridges, peaks, and valleys – an appearance very similar to a mountain range. Three long diffraction spikes from the top right edge of the image suggest the presence of a large star just out of view. For more details, download the Text Description."

Charts and data



Alt text + description

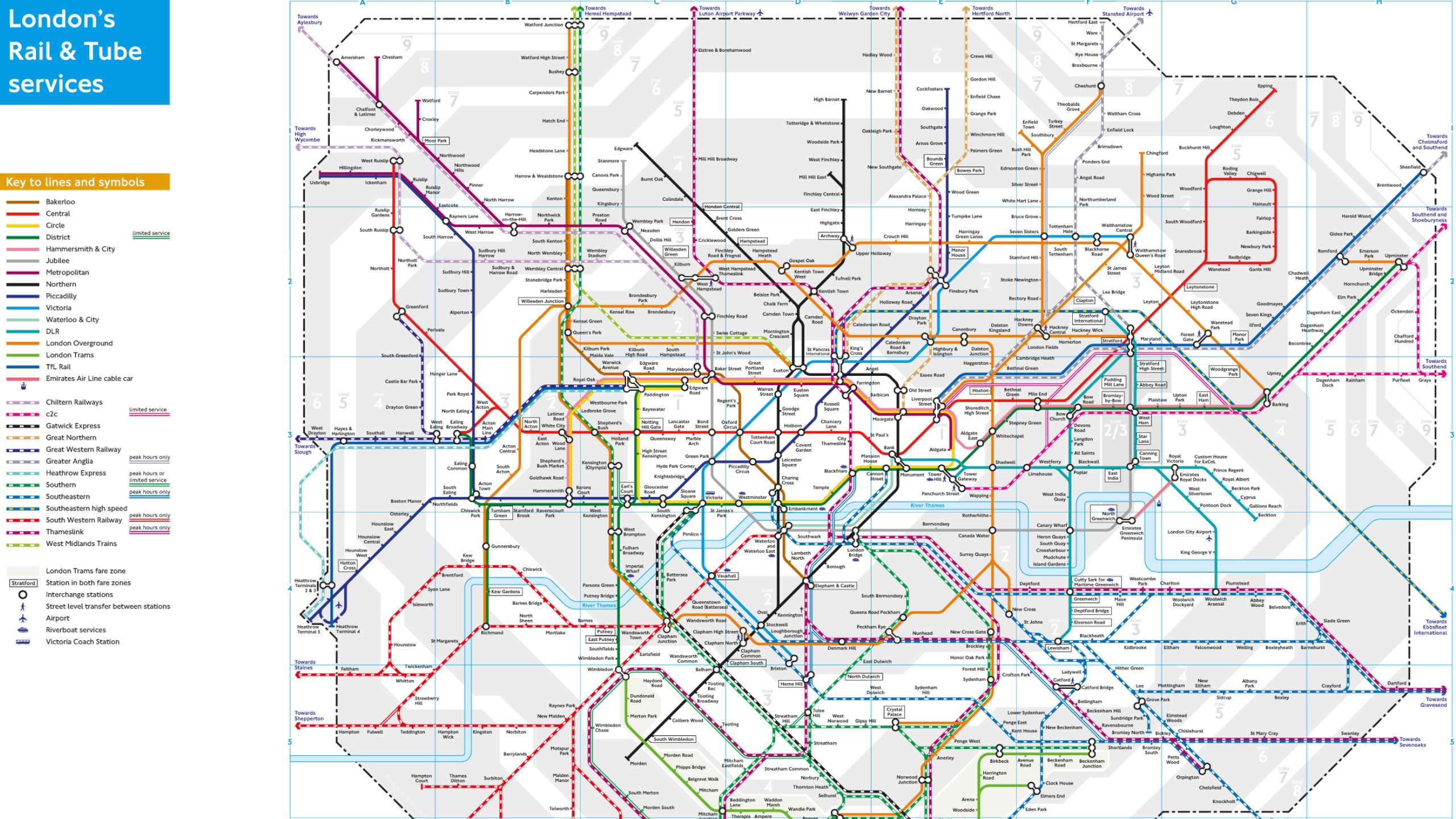
1. Give the image succinct alternative text
2. Provide a detailed description
   - In context **OR**
   - Link to the information



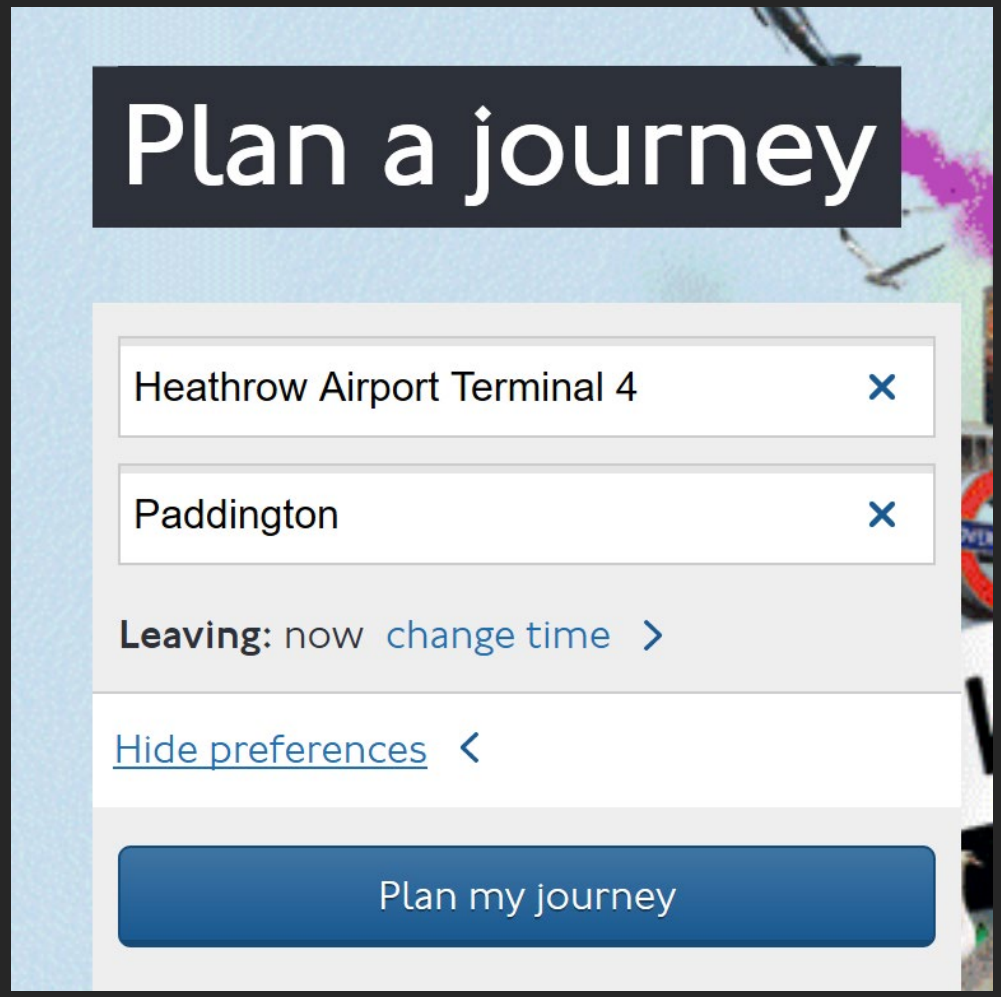
Respondent Region

Region	# of respondents	% of respondents
North America	888	57.7%
Europe	362	23.5%
Asia	126	8.2%
Australia and Oceania	58	3.8%
Africa/Middle East	52	3.4%
South America	42	2.7%
Central America and Caribbean	12	0.8%

London's Rail & Tube services



“Equivalent” experience



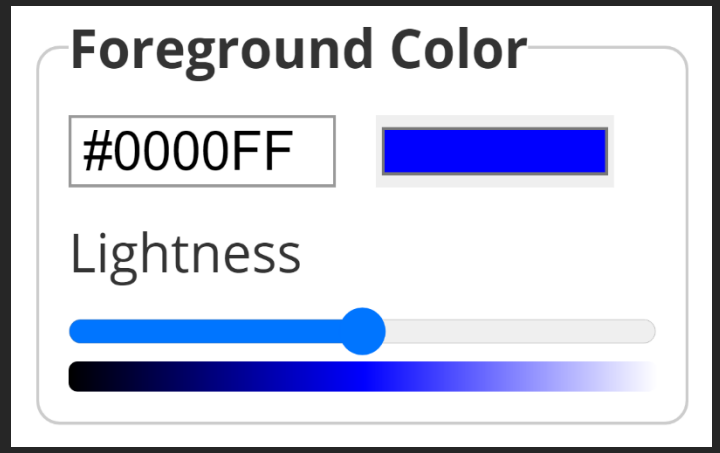
Evaluate alternative text

- Use WAVE
- Search for images that do not have WAVE icons
  - We will cover this later

Forms

HTML Input Types

- Use the correct input type
  - `checkbox` (multiselect)
  - `radio` (single select)
- Native inputs have better accessibility than custom inputs
  - E.g., `color` and `range`
- Mobile devices may present custom keyboards
  - e.g., phone keypad for `type="tel"`





Form Labels

First Name:

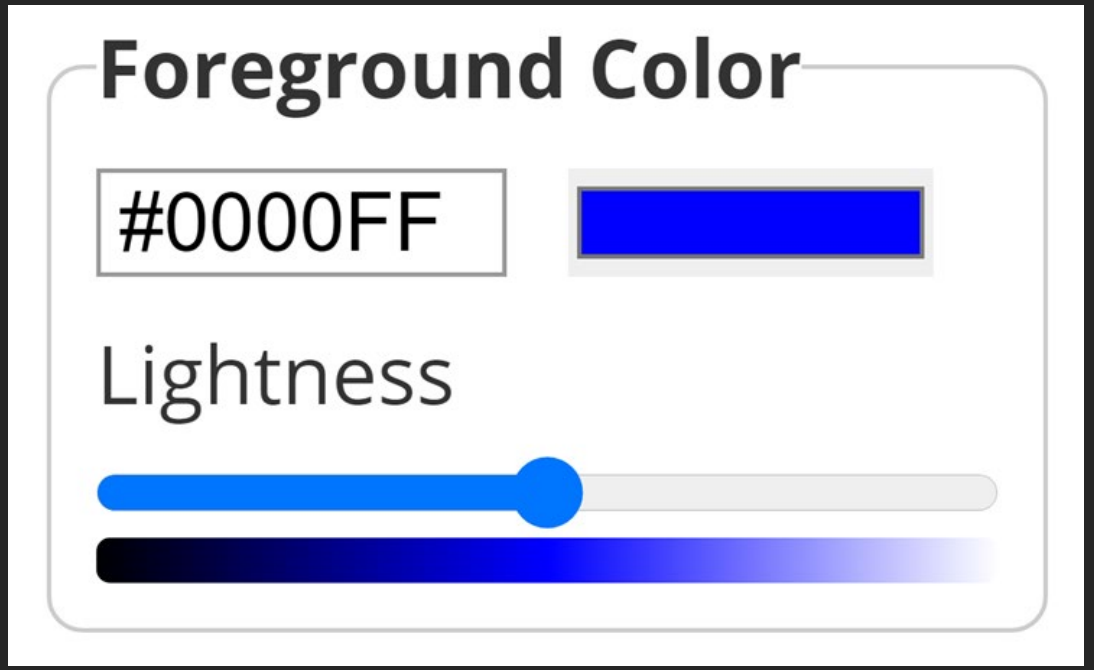
```
<label for="fname">First Name:</label>
<input type="text" id="fname"…>
```

Text boxes, text areas, select menus, checkboxes, radio buttons, and HTML5 input types.

Implicit Form Labels

```
<label>First Name:
<input type="text"></label>
```

<fieldset> and <legend>




```
<fieldset>
  <legend>Foreground Color</legend>
  ...
</fieldset>
```

Evaluate forms and labels

Low Vision

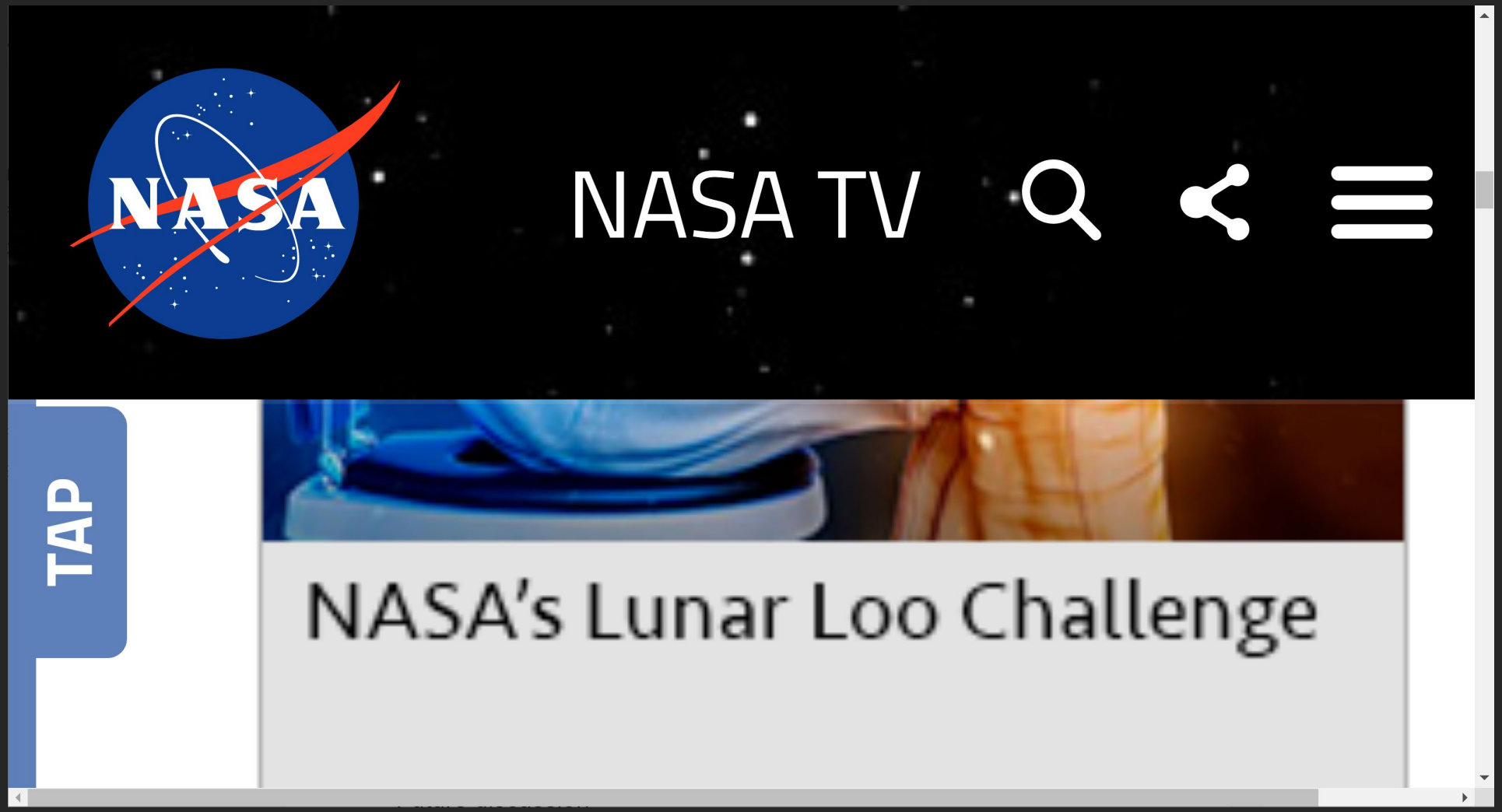
1.4.5 – Images of Text (AA)



Browser Zoom

- Browser controls
  - Larger: Ctrl/command & +
  - Smaller: Ctrl/command & -
  - Reset : Ctrl/command & 0
- Zoomed content will trigger responsive breakpoints
  - Responsive design supports users with low vision who zoom page content

Browser Zoom on nasa.gov



WCAG Zoom Requirements

1.4.4 Resize text (Level AA)

- “Except for captions and images of text, text can be resized without assistive technology up to 200% without loss of content or functionality.”
- Zoom to 200% is supported.

1.4.10 Reflow (Level AA - WCAG 2.1)

- “Content can be presented without loss of information or functionality…” and without horizontal scrolling (except when necessary) at 400% zoom with viewport width at 1280 pixels.

Evaluate page zoom

- Ctrl/command & +, -, 0
- Zoom to 200%
- Resize to 1280px and Zoom to 400%
  - Any horizontal scrolling?
- Test the mobile menu

Contrast and Color

WebAIM contrast article

21:1

21:1

Text Contrast (AA)

4.5:1

Gray on White

Purple on White

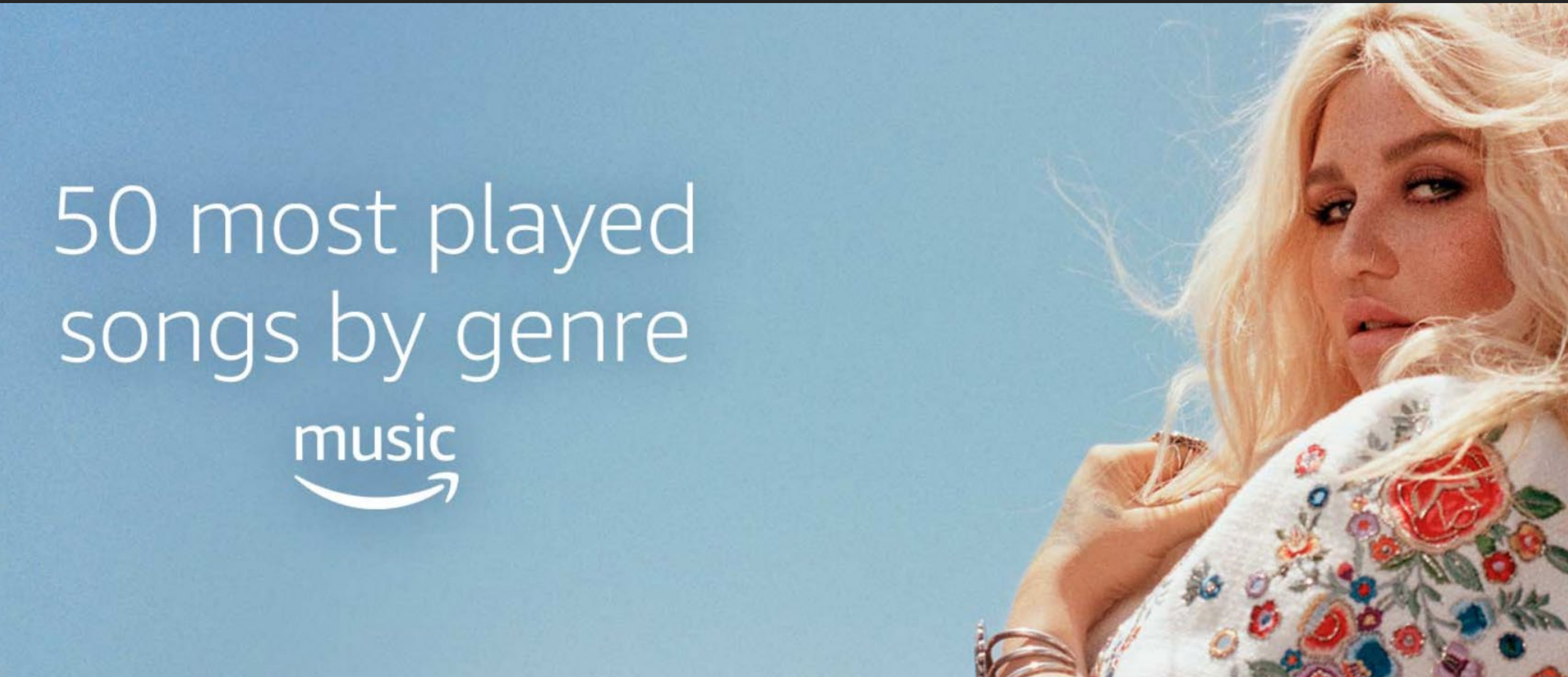
Red on Yellow

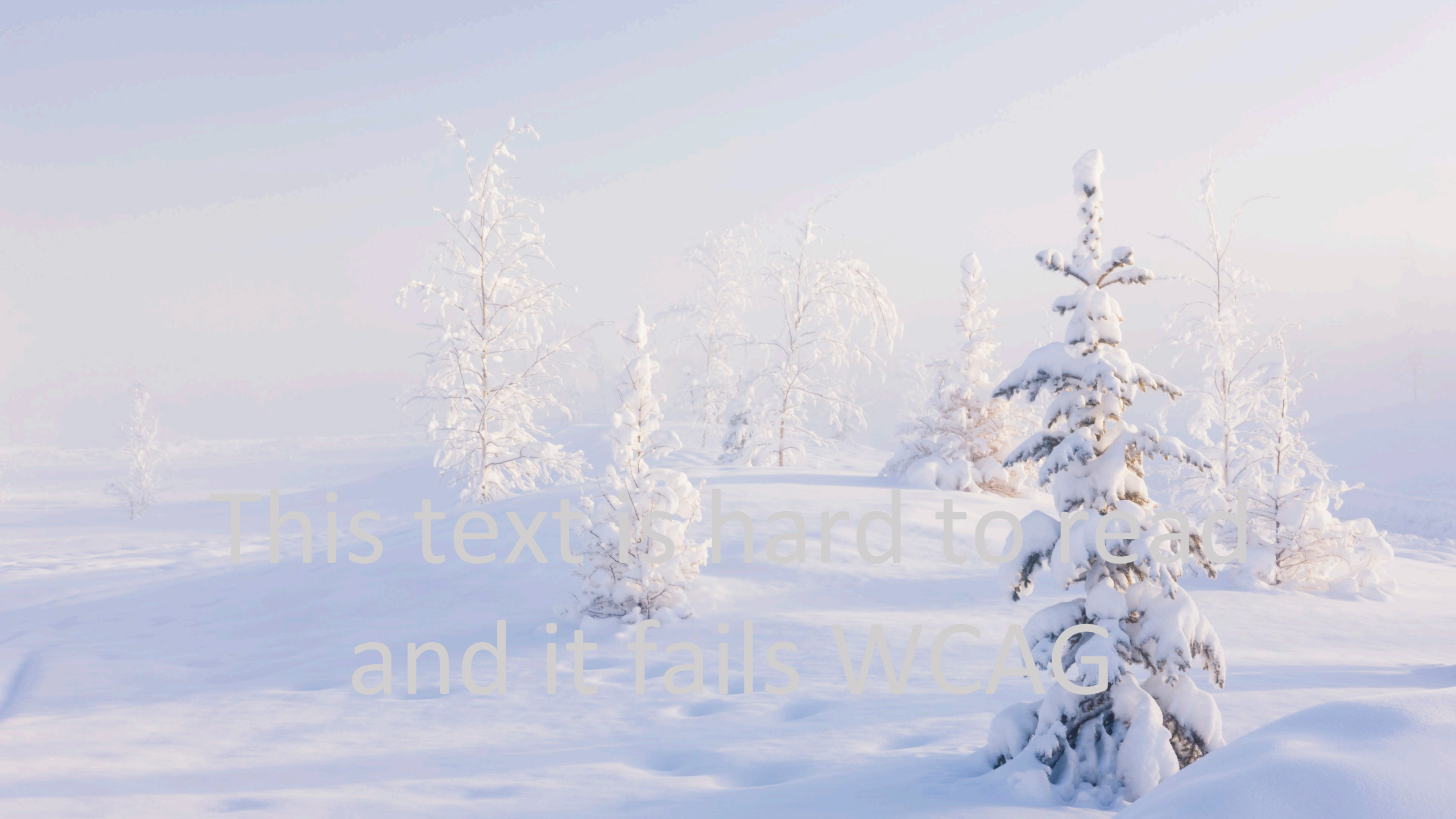
3:1 – “Large Text”

≥18pt (24px)

BOLD: ≥14pt (18.67px)

“Images of text”



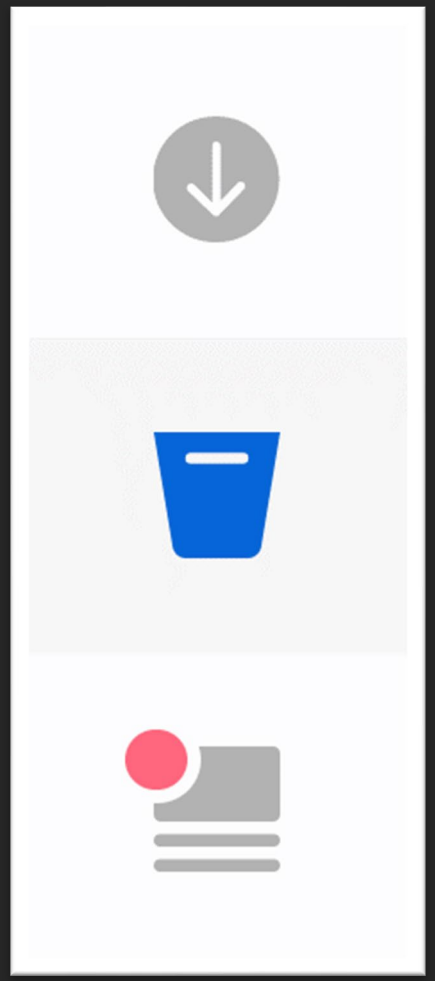

This text is hard to read
and it fails WCAG


This text is hard to read
and it fails WCAG

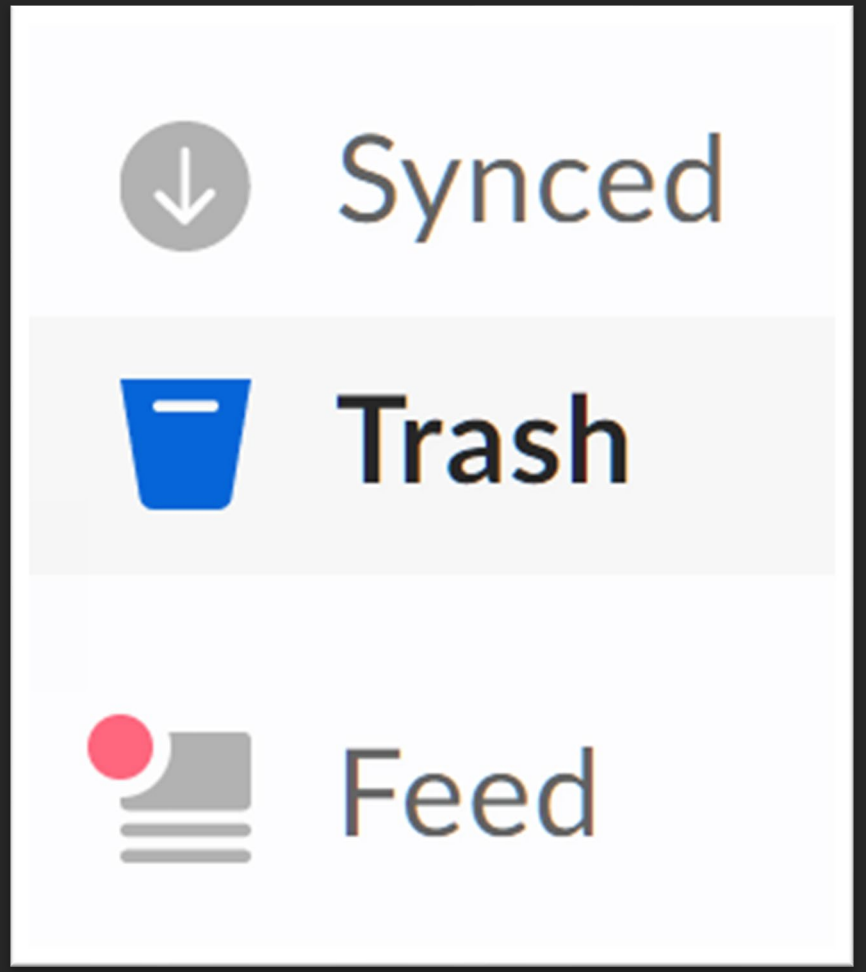
WebAIM Contrast Checker

webaim.org/resources/contrastchecker/

3:1 Contrast of “Graphical Objects”



“…required to understand”



“Against adjacent color(s)"



Add a border



3:1 Contrast of UI “Boundaries”

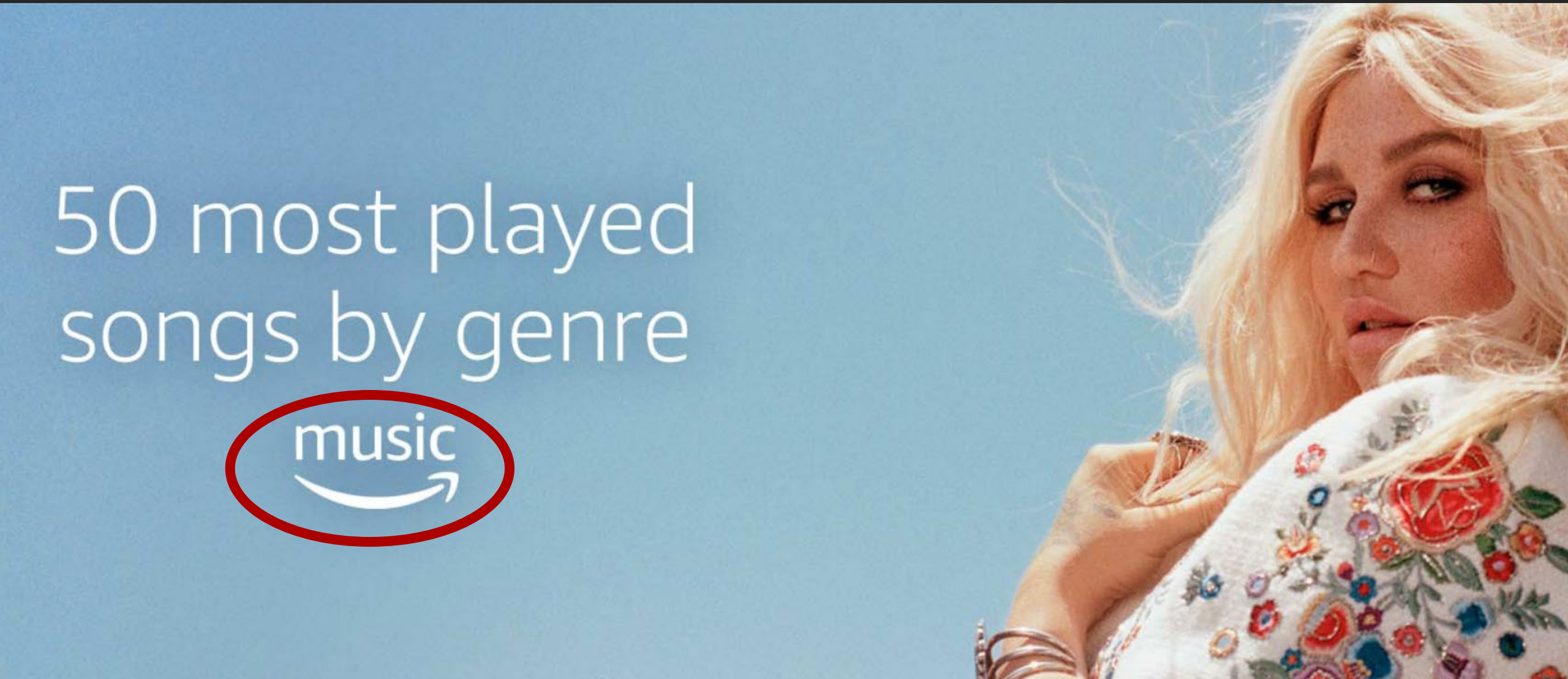


I have read and agree to the Terms of Use

Exceptions

"Inactive"

“Logotypes”



“Pure Decoration”



1.4.1 – Use of Color (A)

The green mushrooms listed here are okay to eat. The red mushrooms are poisonous.

- Amanita
- Chanterelle
- Porcini
- Shiitake
- Tylopilus

How can we fix this?

The green mushrooms listed here are okay to eat. The red mushrooms are poisonous.

- Amanita
- Chanterelle
- Porcini
- Shiitake
- Tylopilus

Icons (with alternative text)

- Amanita 
- Chanterelle
- Porcini
- Shiitake
- Tylopilus 

Separate lists

Edible

- Chanterelle
- Porcini
- Shiitake

Poisonous

- Amanita
- Tylopilus

Bold or Italicized text?

- ***Amanita***
- Chanterelle
- Porcini
- Shiitake
- ***Tylopilus***

Multimedia

Auditory Disabilities



Captioning Methods

- Type, then sync
- Stenographer or “Shadow Speaker”
- Automated (speech recognition)



Other multimedia considerations

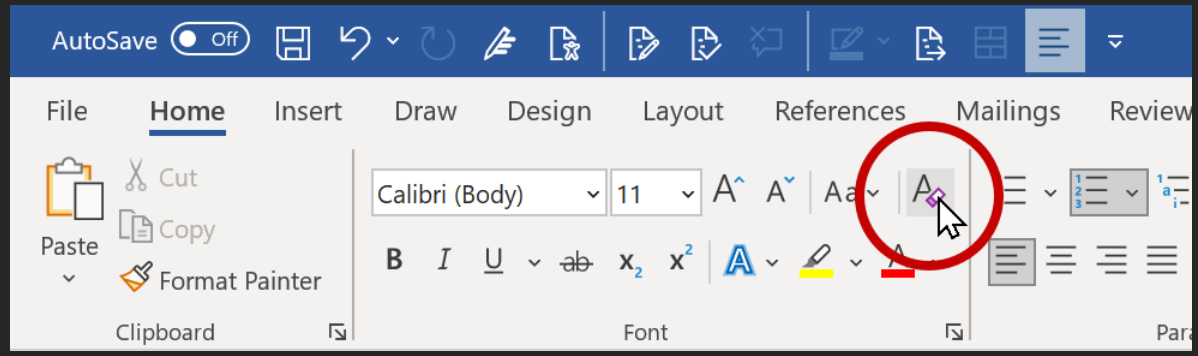
Audio Description

A narrator that describes visual content



Avoid expensive audio descriptions by ensuring important visual content is presented audibly.

Action + Name + Description + Location



1. Action: “Click”
2. Name: “Clear All Formatting”
3. Description: “Letter A with an eraser”
4. Location: “In the Font area of the Home tab”

Tips for Accessible Screen Sharing

Descriptive Transcript



- Called "Alternative for Time-based Media" in WCAG
- Text-based presentation of important audio and visual information
  - Does not have to be text-only
- Not synchronized

Deaf-blind

- Content accessible to blind users will generally be accessible to users who are deaf-blind
  - With the exception of media
- Descriptive transcripts provide media accessibility



WCAG Requirements

Level A

- Captions
- Transcript for audio-only or video-only content
- Audio description OR transcript (if needed)

Level AA

- Audio description (if needed)
- Live audio – Captions

Level AAA

- Descriptive transcript
- Sign language
- Live audio – Transcript

Our recommendation

Level A

- Captions
- Transcript for audio-only or video-only content
- Audio description OR transcript (if needed)

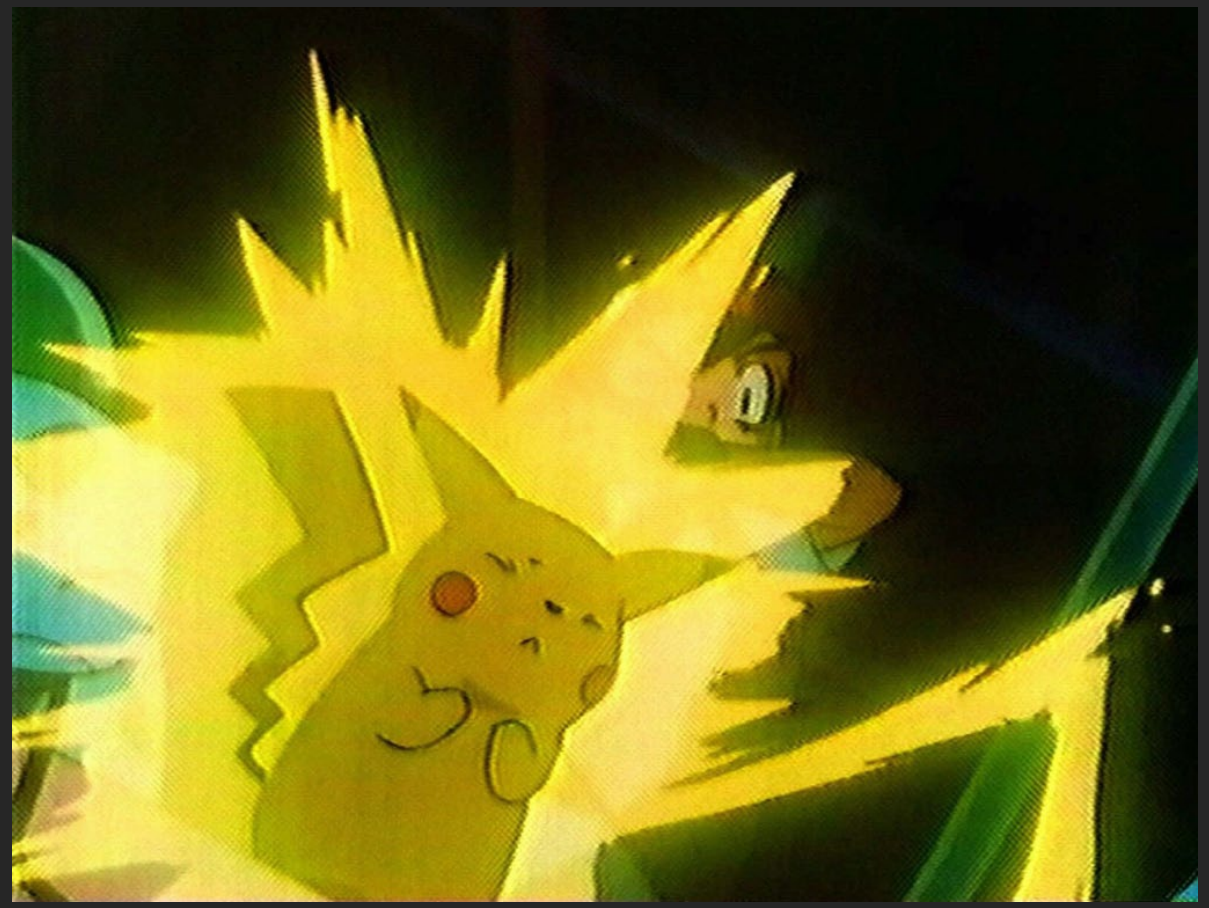
Level AA

- Audio description (if needed)
- Live audio – Captions
- **Descriptive transcript**

Level AAA

- Sign language
- Live audio – Transcript

Photosensitive Epilepsy



Caution with flashing/strobing content

- 3 times per second or greater
- Size, brightness, and red threshold
- “Annoying rule”

Cognitive/Learning Disabilities

- Largest disability group.
- Most of the earlier principles can make content more understandable.
  - Captions, contrast, color, headings, lists, simple tables, etc.
- Because users' needs vary greatly, we will focus on general recommendations.

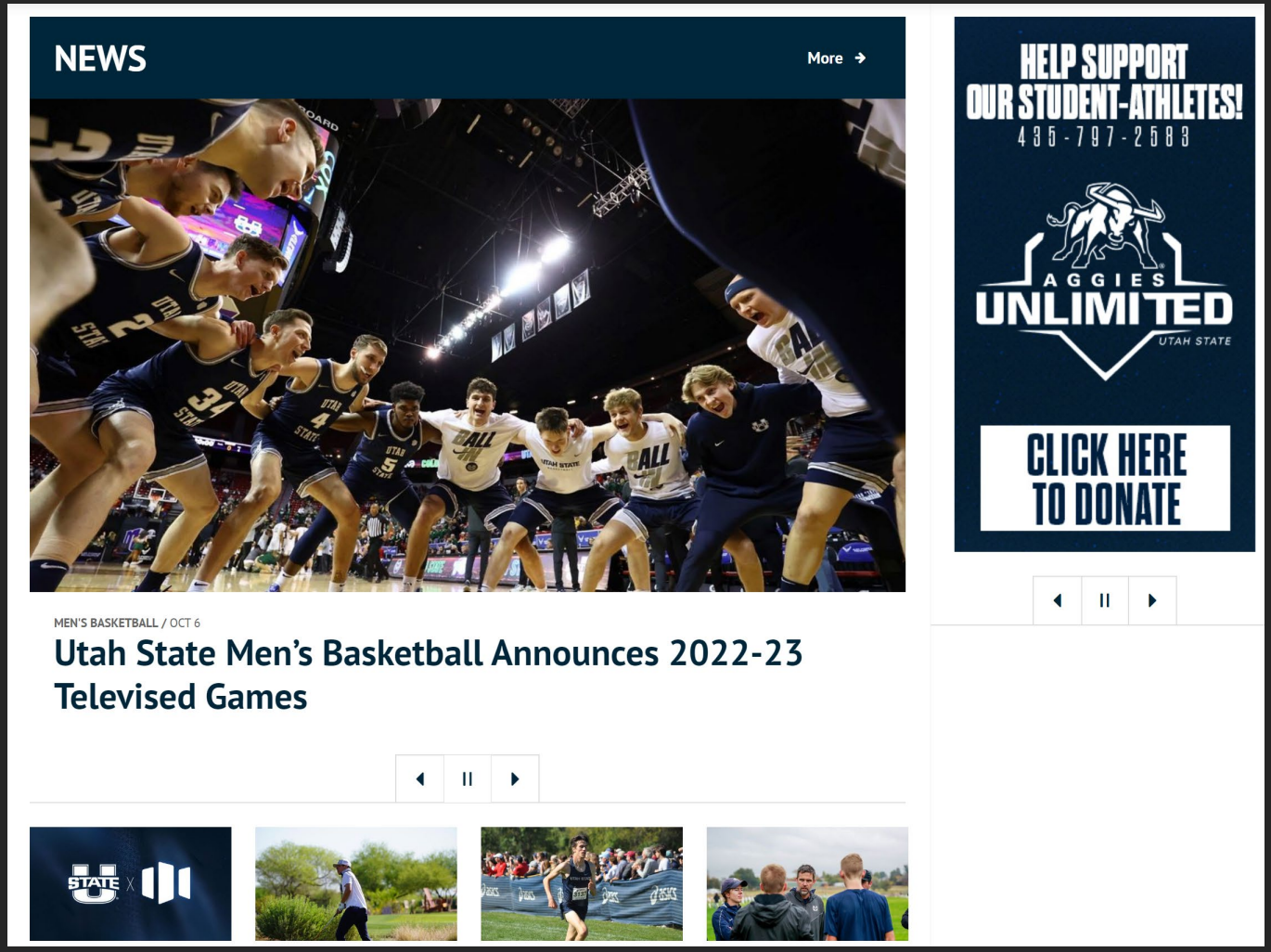
Make Content “Understandable”

- Be consistent
  - “Consistent Navigation”, “Consistent Identification”
    - “Consistent Help” (WCAG 2.2)
- Use plain language
  - “readable to users with a lower secondary reading level” (WCAG AAA)
- Write for your audience
- Be careful with movement and other distractions

There must be a way to pause/stop/hide moving content that:

1. Starts automatically
2. lasts more than five seconds,
3. *and* is presented in parallel with other content.

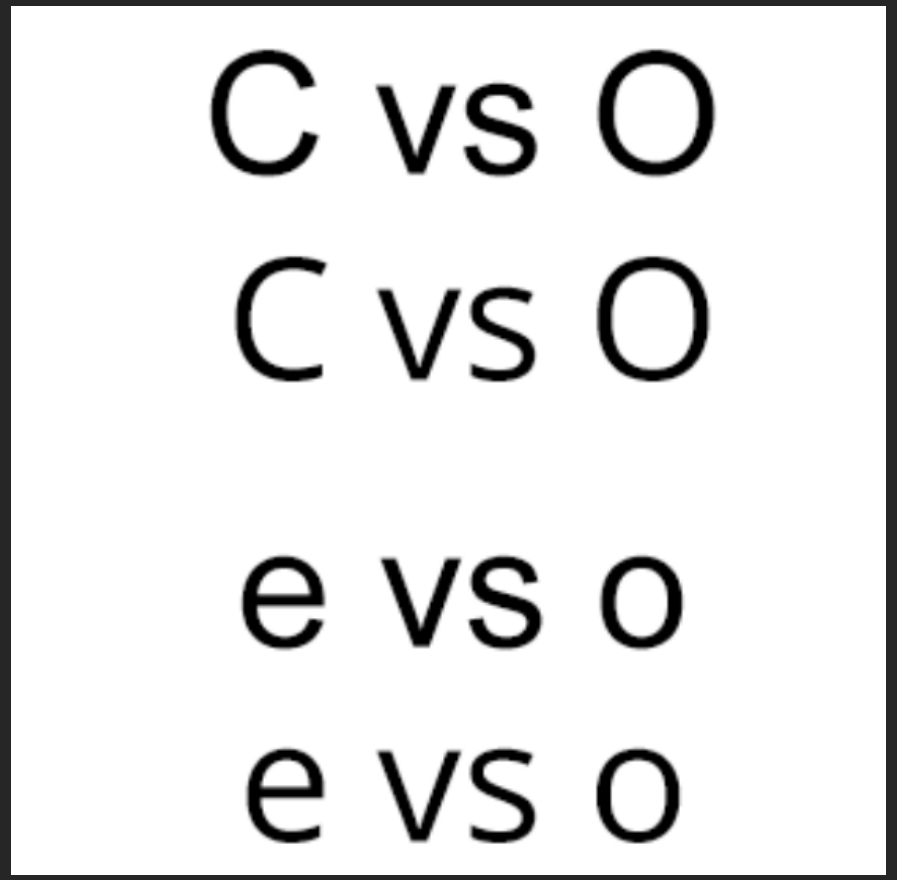
E.g., Carousels



Legible Text

- Small text negatively impacts readability
  - WCAG has no text size requirements
- Avoid long line lengths
  - Consider line height/spacing
- Choose legible fonts

Typefaces and Fonts







1.4.12 Text Spacing (Level AA)

“No loss of content or functionality occurs” when the ***user*** increases spacing between:

- Paragraphs: 2 × font size
- Lines: 1.5 × font size
- Words: 16% font size
- Letters: 12% font size

Text Spacing Applied

Earlier guidelines [edit]

The first web accessibility guideline was compiled by Gregg Vanderheiden and released in January 1995, just after the 1994 Second International Conference on the World-Wide Web (WWW II) in Chicago (where Tim Berners-Lee first mentioned disability access in a keynote speech after seeing a pre-conference workshop on accessibility led by Mike Paciello).[2]

Over 38 different Web access guidelines followed from various authors and organizations over the next few years.[3] These were brought together in the *Unified Web Site Accessibility Guidelines* compiled at the University of Wisconsin-Madison.[4] Version 8 of the *Unified Web Site Accessibility Guidelines*, published in 1998, served as the starting point for the W3C's WCAG 1.0.[5]

WCAG 1.0 [edit]

The WCAG 1.0 was published and became a W3C recommendation on 5 May 1999. They have since been superseded by WCAG 2.0.

WCAG 1.0 consist of 14 guidelines—each of which describes a general principle of accessible design. Each guideline covers a basic theme of web accessibility and is associated with one or more checkpoints that describes how to apply that guideline to particular webpage features.

- Guideline 1: Provide equivalent alternatives to auditory and visual content
- Guideline 2: Don’t rely on colour alone
- Guideline 3: Use markup and style sheets, and do so properly
- Guideline 4: Clarify natural language usage
- Guideline 5: Create tables that transform gracefully
- Guideline 6: Ensure that pages featuring new technologies transform gracefully
- Guideline 7: Ensure user control of time sensitive content changes
- Guideline 8: Ensure direct accessibility of embedded user interfaces
- Guideline 9: Design for device independence

Earlier guidelines [edit]

The first web accessibility guideline was compiled by Gregg Vanderheiden and released in January 1995, just after the 1994 Second International Conference on the World-Wide Web (WWW II) in Chicago (where Tim Berners-Lee first mentioned disability access in a keynote speech after seeing a pre-conference workshop on accessibility led by Mike Paciello).[2]

Over 38 different Web access guidelines followed from various authors and organizations over the next few years.[3] These were brought together in the *Unified Web Site Accessibility Guidelines* compiled at the University of Wisconsin-Madison.[4] Version 8 of the *Unified Web Site Accessibility Guidelines*, published in 1998, served as the starting point for the W3C's WCAG 1.0.[5]

WCAG 1.0 [edit]

The WCAG 1.0 was published and became a W3C recommendation on 5 May 1999. They have since been superseded by WCAG 2.0.

WCAG 1.0 consist of 14 guidelines—each of which describes a general principle of accessible design. Each guideline covers a basic theme of web accessibility and is associated with one or more checkpoints that describes how to apply that guideline to particular webpage features.

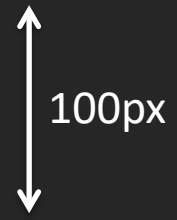
- Guideline 1: Provide equivalent alternatives to auditory and visual content
- Guideline 2: Don’t rely on colour alone
- Guideline 3: Use markup and style sheets, and do so properly
- Guideline 4: Clarify natural language usage

Evaluate text spacing

Add text spacing bookmarklet

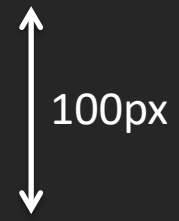
Avoid CSS `height`

I am some text in a div that has a pixel height



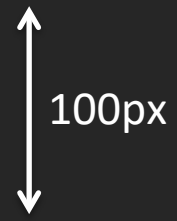
Avoid CSS `height`

I am some text in a div that has a pixel height



Use CSS `min-height`

I am some text in a div that has a pixel min-height



Advanced Contrast

How many contrast and color checks?

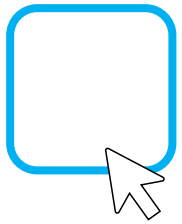


I have read and agree to the Terms of Use

15 or more

Evaluating Color and Contrast – How hard can it be?

You must test contrast of different states



I have read and agree to the Terms of Use



I have read and agree to the Terms of Use



WCAG 2.0 – Keyboard focus is visible

Avoid

```
a {outline:0;}
```

or

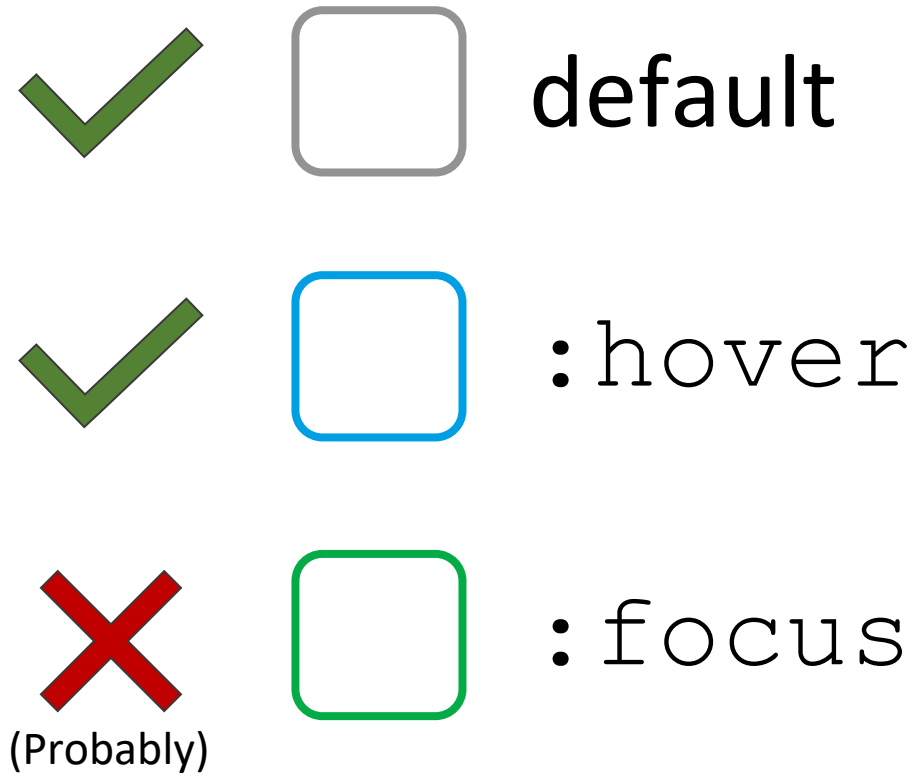
```
a {outline:none;}
```

WCAG AA Failure; no minimum contrast

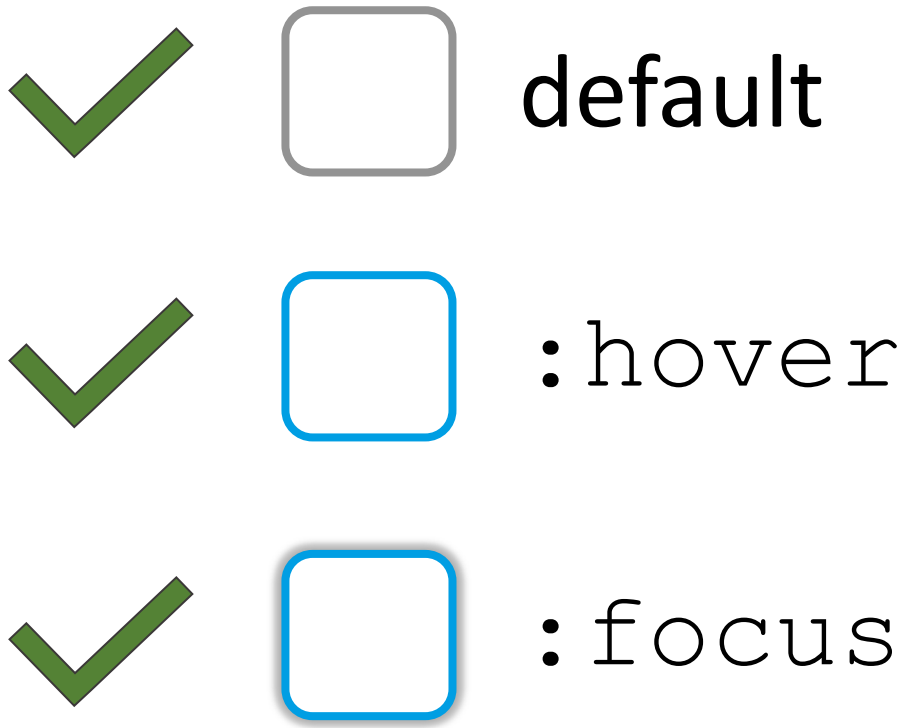
WCAG 2.1 – 3:1 contrast of keyboard focus indicators



3:1 between focused and unfocused state



Gap between “Use of Color” and “Non-Text Contrast”



default

:hover

:focus

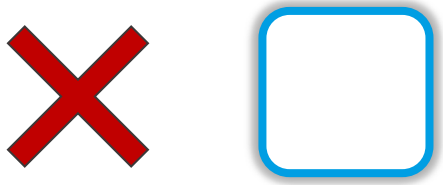
WCAG 2.2 – 2.4.13 Focus Appearance

- Size and contrast requirements for keyboard focus
- Level AAA, but important

At least 3:1 contrast “between the same pixels in the focused and unfocused states”

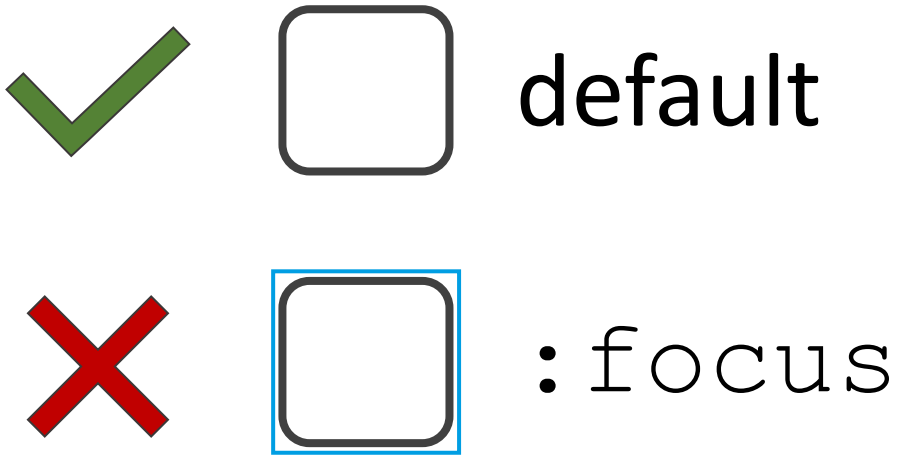


default

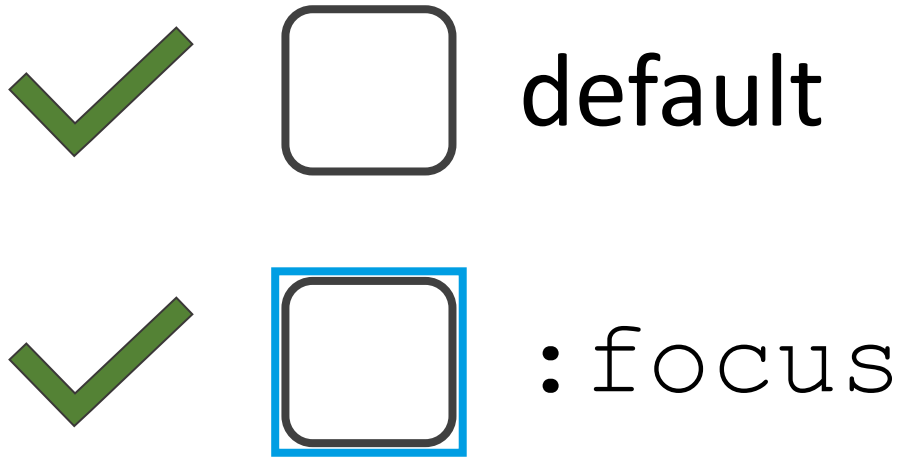


`:focus`

AND at least 2 CSS pixel “perimeter”



Address with a clear keyboard focus outline



WCAG 2.2

- 2.4.11 Focus Not Obscured (Minimum) (AA)
  - When a user interface component receives keyboard focus, the component is not ***entirely hidden*** due to author-created content.
- 2.4.12 Focus Not Obscured (Enhanced) (AAA)
  - When a user interface component receives keyboard focus, ***no part of the component is hidden*** by author-created content.

1. Ensure sufficient contrast
2. Do not rely on color alone

You must agree to the Terms of Use

You must agree to the Terms of Use

Requirements for links that rely on color

1. A 3:1 contrast ratio between link text and non-link text
2. Link must present a non-color cue (typically underline) on mouse hover AND keyboard focus
   - Not possible in most electronic documents (e.g., Word and PowerPoint)

You must agree to the Terms of Use

webaim.org/resources/linkcontrastchecker

You must agree to the Terms of Use

You must agree to the Terms of Use

and link becomes underlined on hover/focus.

You must agree to the **Terms of Use**

You must agree to the **Terms of Use**

You **must** agree to the **Terms of Use**

You must agree to the Terms of Use

Motor Disabilities

“The myth of the keyboard-only user”

Joe Dolson

Pointing Devices & Voice Control

- Mouse users and touchscreen users
  - Less dexterity
  - Tremors
  - Chronic pain
- Assistive technologies that behave like a mouse
  - E.g., Eye tracking
- Voice Control

Target Size Requirements

- Ensure adequate target size and spacing between clickable elements
  - 2.5.5 Target Size (WCAG 2.1 - Level AAA) requires 44X44 pixel targets
  - WCAG 2.2 adds “Target Size (Minimum)” at AA
    - a 24X24 pixel target size
    - If 24 pixel circles are centered on each interactive element, no circles intersect.


9:41
FaceTime
Monday
3
Calendar
Photos
Camera
Mail
Clock
Maps
Weather
Reminders
Notes
Stocks
News
Books
App Store
Podcasts
tv
TV
Health
Home
Wallet
Settings

!= Touch Only

Keyboard users

- Many screen reader users
- Some users with low vision
- Users with motor disabilities who prefer a keyboard
  - Using a pointing device may is possible, but it is less efficient, more difficult, or painful
- Power users

Use Standard Keyboard Commands

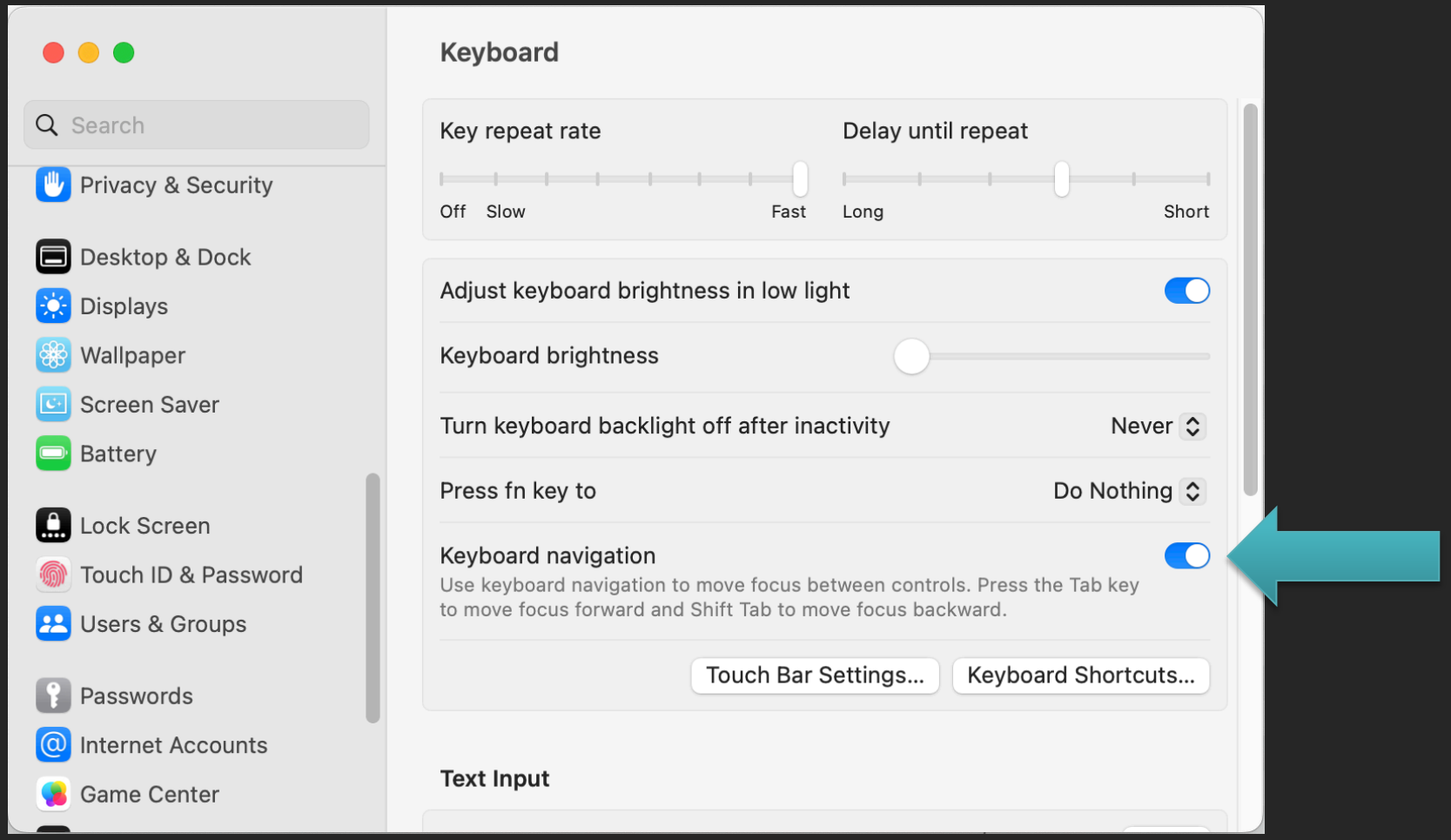
- Navigate links, form controls, etc.: Tab, Shift + Tab
- Link: Enter
- Button: Enter **or** Spacebar
- Toggle (e.g. checkbox): Spacebar
- Navigate options in one “tab stop”: ↑/↓ or ←/→
- Close/Collapse: Esc
- Scroll the page (unless focused on a control): Spacebar

E.g., WebAIM Form Controls article

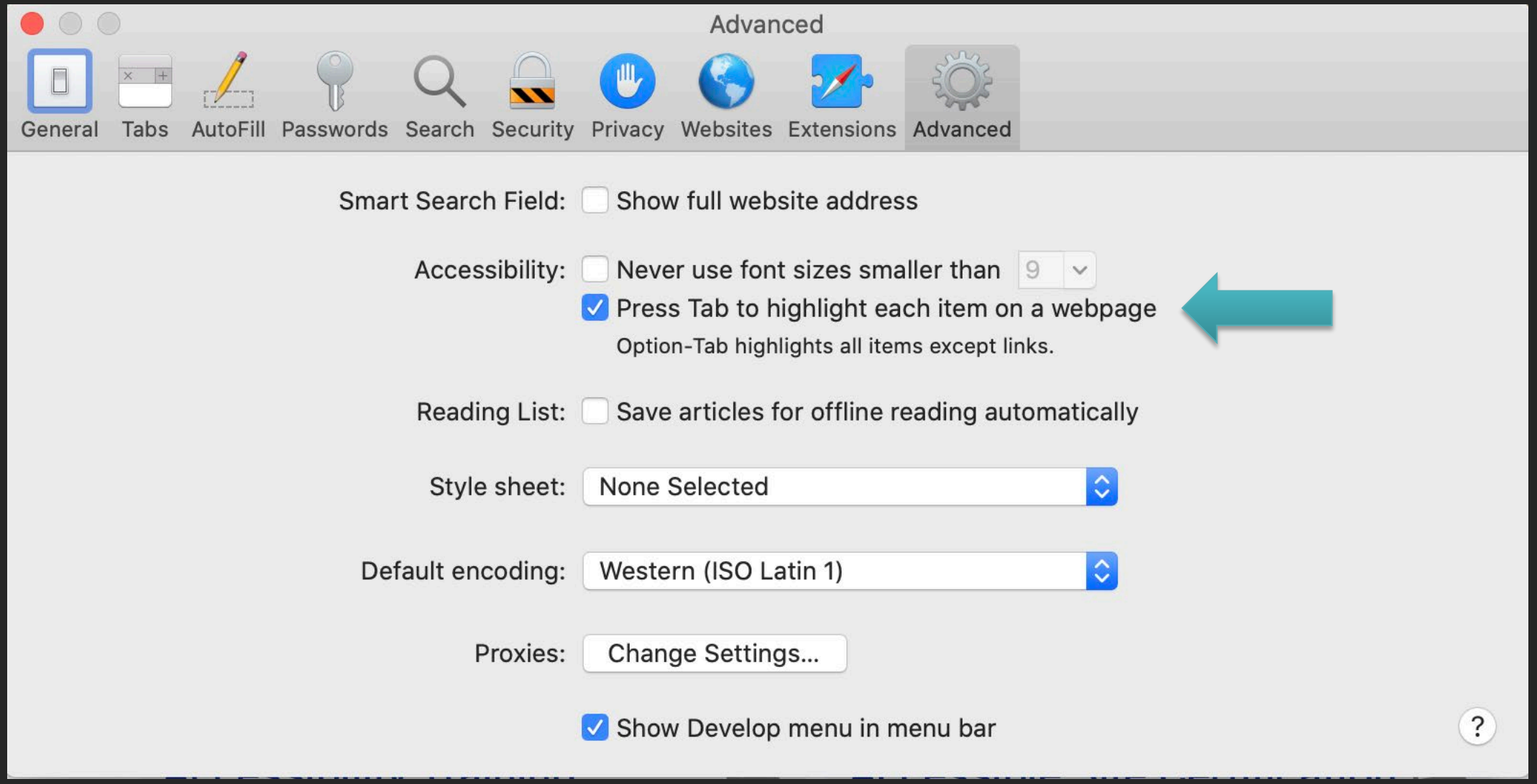
Follow Design Patterns for Custom Widgets

ARIA Authoring Practices Guide (APG)

Some tweaking required on Mac



Safari Keyboard Settings



HTML elements are
natively keyboard accessible

…unless you break things with CSS or JavaScript

Use the Correct Element

- Links to open a new page or jump to another location in the page
- Buttons to submit form data or perform an in-page function
- Other form controls (e.g., `<input>`) to enter information

Hiding Content

- Hide from everyone:
  - `display:none`
  - `visibility:hidden`
  - HTML `hidden` attribute
- An element should never be hidden while it has keyboard focus.
  - Use `hidden` or `display:none` **OR**
  - Make visible when it receives focus

Visually Hiding Content

Position off-screen left with CSS for screen readers

```
.sr-only {
    position:absolute;
    left:-10000px;
    top:auto;
    width:1px;
    height:1px;
    overflow:hidden;
}
```

Can also use CSS clip and/or clip-path

“Skip” links

```
<a href="#maincontent">Skip to
main content</a>

…

<main id="maincontent">...
```

WCAG 2.4.1 (Level A): A mechanism is available to bypass blocks of content that are repeated on multiple Web pages.

Hidden “Skip” links

```
a#skip {
  position:absolute;
  left:-10000px;
  top:auto;
  width:1px;
  height:1px;
  overflow:hidden;
}

a#skip:focus {
  position:static;
  width:auto;
  height:auto;
}
```

Don't use `display:none` on the link

Evaluate keyboard accessibility

Navigate the site using only the keyboard (Tab, Shift + Tab, Enter, Space, ↑/↓ or ←/→, Esc).

- Is there a clear keyboard focus indicator?
- Is all functionality available?
  - How about responsive layouts?
- Is navigation order logical?
  - WAVE “Order” tab

“Label” and “Name” in WCAG

- The **Label** is visually presented
- The **Name** is presented to assistive technology
  - Also called “accessible name”
  - May be visually hidden

3 principles of accessible names

1. All interactive elements must have an accessible name.
2. An element can only have one accessible name.
3. To be sure of the name, you must test with a screen reader, use the WAVE “Order” tab, or inspect the HTML.

What is the WCAG “Label”?



What is the WCAG “Name”?




```
<a href="page2.php"…>
   <img alt="Next"…>
</a>
```

“Label” and “Name” are often the same

Code	Label	Name
`<a href="…">Log in</a>`	Log in	“Log in”
`<label>First Name: <input type="text"> </label>`	First Name:	“First Name:”
`<button>Activate </button>`	Activate	“Activate”

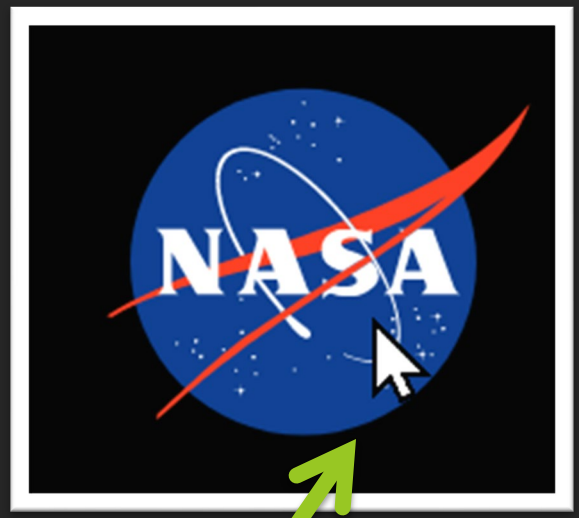
2.5.3 – Label in Name (Level A)

“For user interface components with labels that include text or images of text, the name contains the text presented.”

“User interface components”



UI component



NOT a UI component

“Labels that include text or images of text”



Text in label

No text in label

“The name contains the text presented”

Label not in Name – Fails 2.5.3




```
<a href="page2.php"…>
    <img alt="Continue"…>
</a>
```

Label *in* Name – Passes 2.5.3




```
<a href="page2.php"…>
    <img alt="Next Page"…>
</a>
```

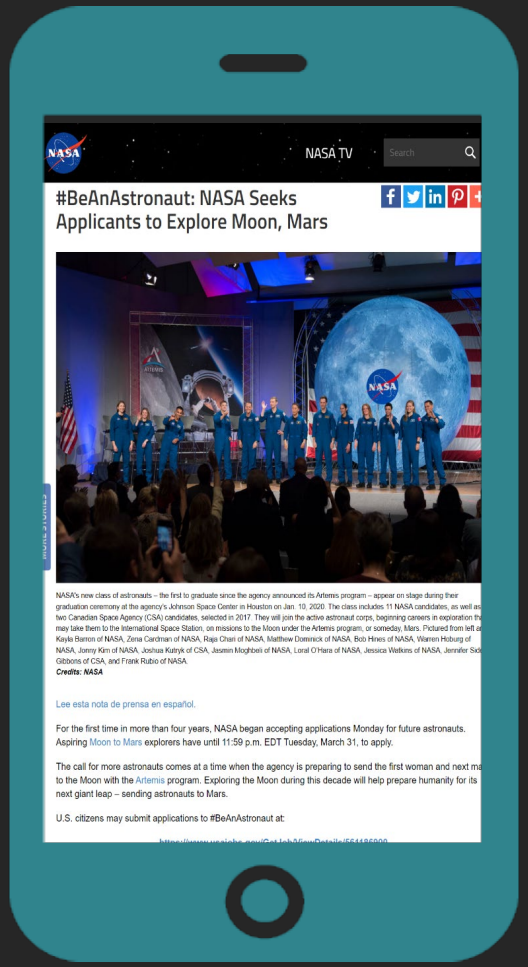
Better solution

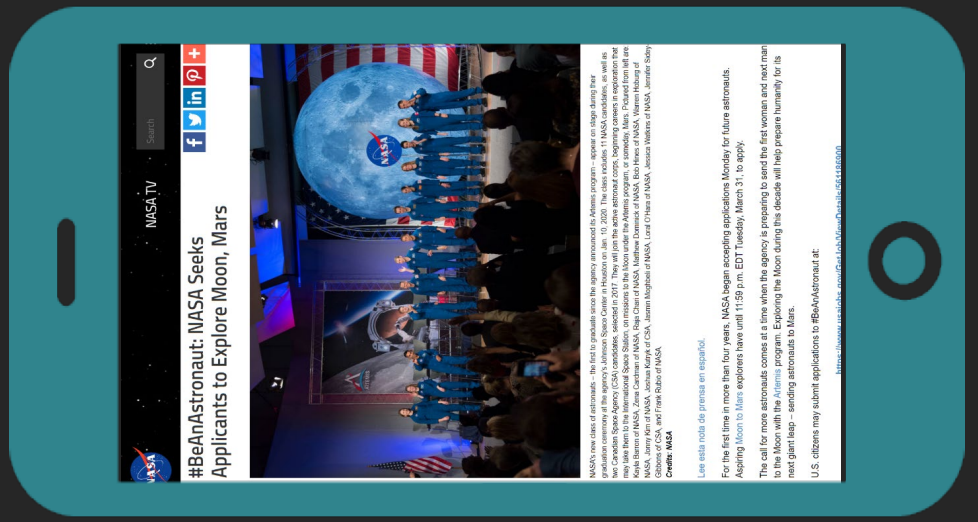



```
<a href="page2.php"…>
   <img alt="Next"…>
</a>
```

WCAG 2.1 - Mobile

1.3.4 (AA) – Don’t lock orientation





WCAG 2.1 – Guideline 2.5 Input Modalities

- 2.5.1 (A) – Don’t rely on multipoint or path-based gestures (e.g., swipe to navigate)
  - 2.5.7 (AA, WCAG 2.2) – Dragging movement has a single pointer alternative
- 2.5.2 (A) – Pointer functionality can be canceled (e.g., no down-event to trigger)
- 2.5.4 (A) – Don’t rely on device motion (e.g., shake, tilt, or pan)

Rule #1 of ARIA Use

If you can use HTML, then do so.

Rule #2

Do not change native semantics,
unless you really have to.

```
<div role="navigation">
  <ul>…</ul>
</div>
```

instead of

```
<ul role="navigation">
```

HTML and ARIA Roles

- HTML elements have default roles
- An element can only have 1 role
- ARIA `role` overrides the default HTML role

ARIA Roles

- Do not change browser behavior
- Can change screen reader behavior
- Can destroy accessibility when misapplied



```
<input type="checkbox" role="radio">
```

ARIA states and properties

- Elements can have more than one `aria-` attribute.
- ARIA states and properties should not override HTML properties if there's a conflict—usually.
- Be very careful!



```
<input type="checkbox" aria-checked="false">
```

Rule #3

All interactive ARIA controls must be usable with the keyboard.

Design Patterns

ARIA Authoring Practices Guide (APG)

```
<div onclick="submitForm()">
Search</div>
```

Tabindex of 1 or greater

- `tabindex="1+"` defines an explicit tab order
- Never use positive tabindex values!
- If the default tab order is not logical, fix your source code order.

Tabindex of 0 and -1

- `tabindex="0"` allows things besides links and form controls to receive keyboard focus.
- `tabindex="-1"` allows things to receive programmatic focus (by scripting, links, etc.)
  - Necessary for focusing dialog boxes, error messages, etc.
  - WARNING: This removes the element from the default tab order.
- `tabindex 0/-1` also used together to manage keyboard focus in some widgets (e.g., a tree menu or tab panel)

```
<div onclick="submitForm()"
tabindex="0">
Search</div>
```

Check for Enter (13) and Space (32) key presses

```
if(event.keyCode==13 || event.keyCode==32)
{
    doStuff();
}
```

```
<div onclick="submitForm()"
tabindex="0">
Search</div>
```

```
<a href="#" onclick="submitForm()">
Search</a>
```

```
<a href="#" onclick="submitForm()"
role="button">
Search</a>
```

Just Use A Button

```
<button onclick="submitForm()">
        Search</button>
```

Rule #4

Do not use `role="presentation"` or `aria-hidden="true"` on visible, focusable elements.

Rule #5

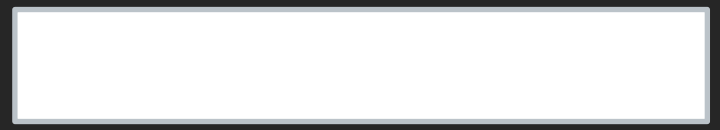
All interactive elements must have an accessible name

What if you have more than one candidate for the accessible name?

1. If the control has an `aria-label` or an `aria-labelledby` attribute the a
   using the algorithm defined in Accessible Name and Description: Compu
2. Otherwise use the associated `label` element(s) *accessible name*(s) - if
   concatenate by DOM order, delimited by spaces.
3. If the *accessible name* is still empty, then: use the control's `title` attribu
4. Otherwise use the control's `placeholder` attribute.
5. If none of the above yield a usable text string there is no *accessible nam*

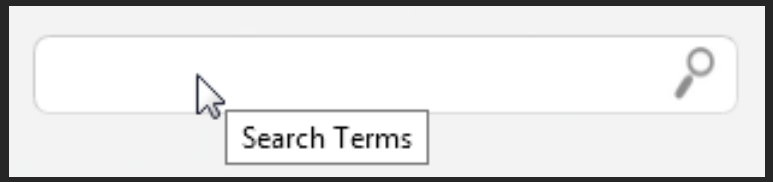
Accessible Name Computation

Form `<label>`

First Name: 

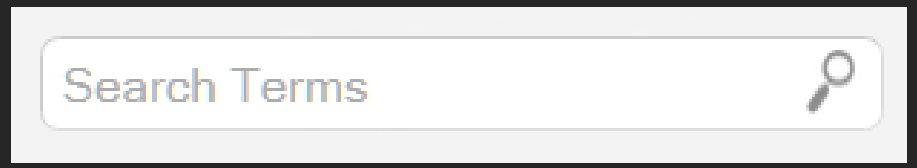
```
<label for="fname">First Name:</label>
<input type="text" id="fname"…>
```

`title` attribute (if no `<label>`)




```
<input title="Search Terms"…>
```

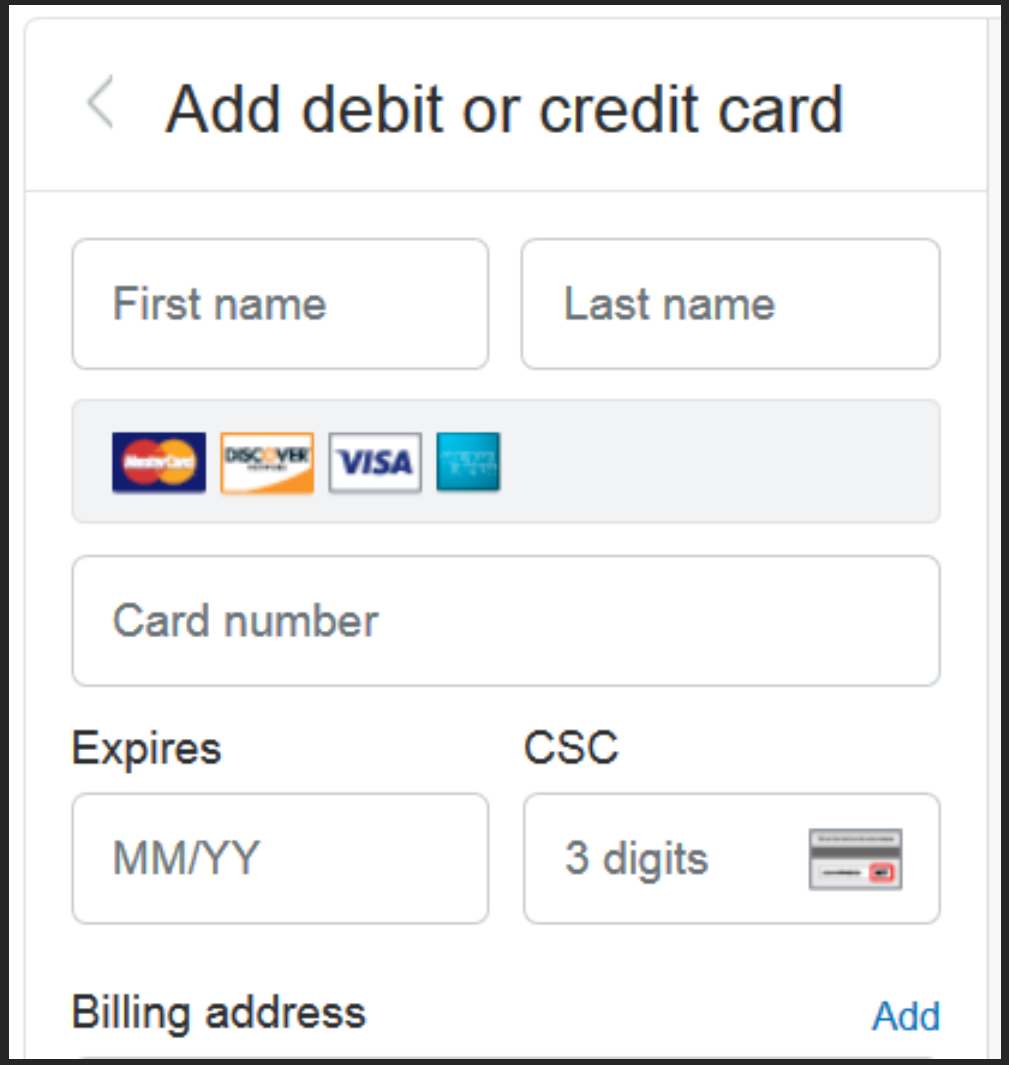
`placeholder` attribute (if no `title` or `<label>`)

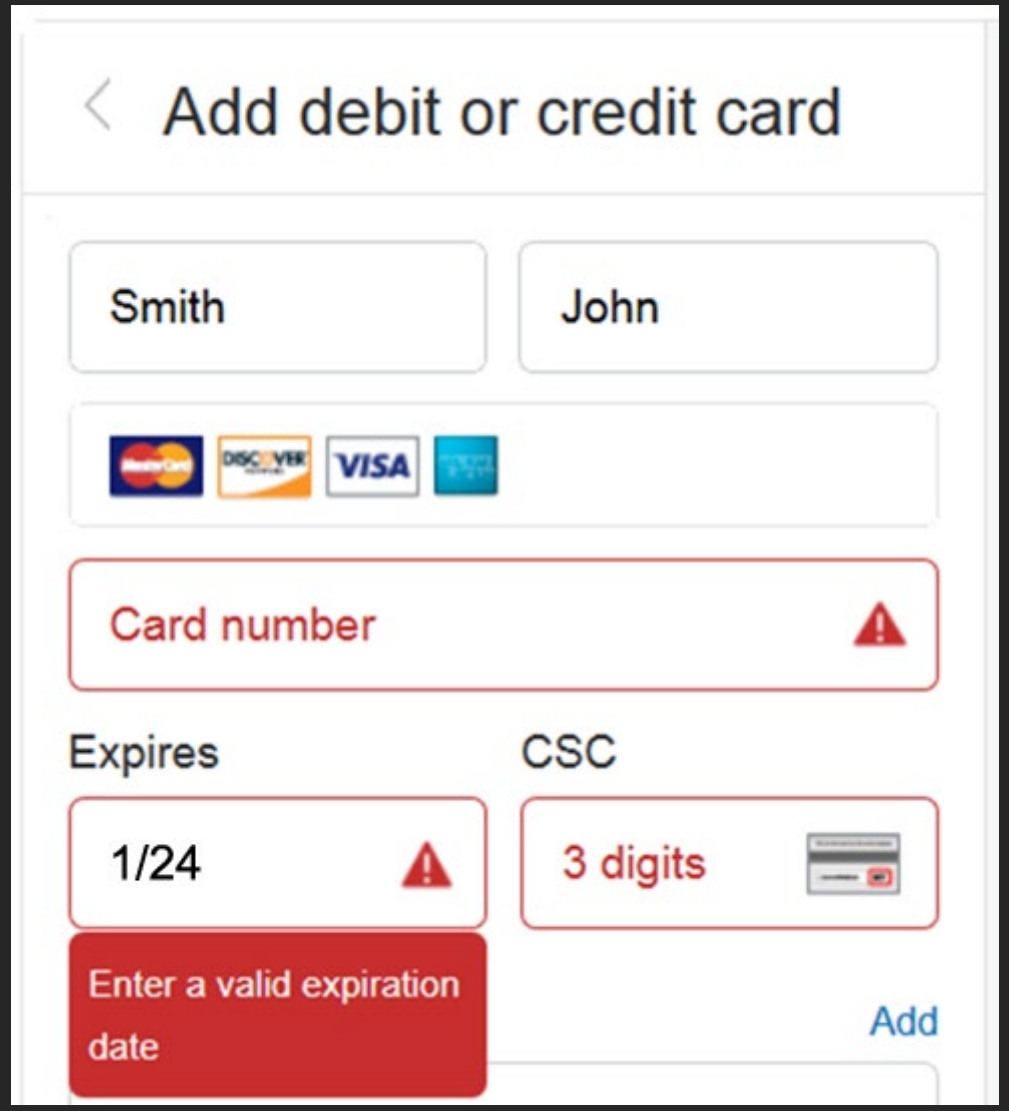


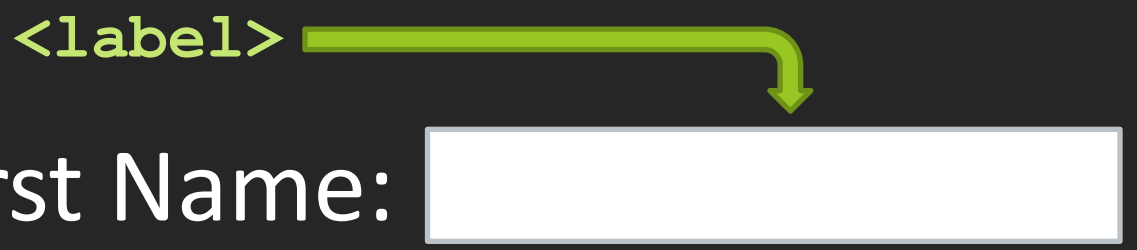

```
<input placeholder="Search Terms"…>
```

Placeholder is read by a screen reader (accessible name),
but is not a suitable “label”


name@example.com
password
confirm password
first name
last name
birthday


Add debit or credit card
First name
Last name
Card number
Expires
CSC
MM/YY
3 digits
Billing address
Add


Add debit or credit card
Smith
John
Card number
Expires
CSC
1/24
3 digits
Enter a valid expiration date
Add

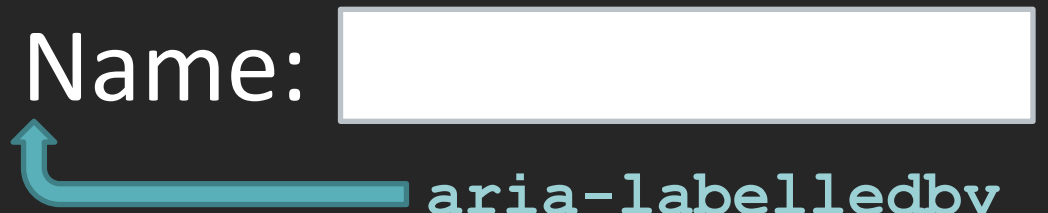



```
<label for="fname">First name:</label>

<input type="text" id="fname">
```

1:1 relationship between `<label>` and a form control.

First Name:




```
<span id="fnamelabel">First name:</span>

<input type="text"
aria-labelledby="fnamelabel">
```



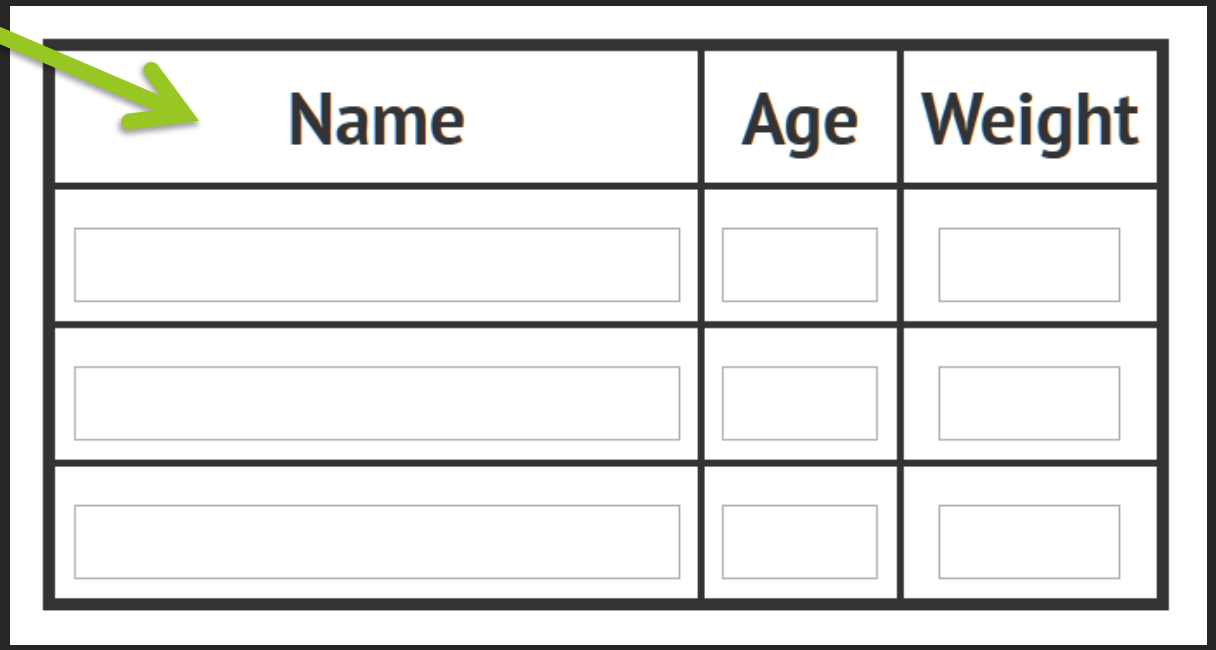

```
<label for="fname" id="fnamelabel">
First name:</label>

<input type="text" id="fname"
aria-labelledby="fnamelabel">
```

Unnecessary markup (Rule #1 of ARIA),
but label will only be read once.

One label for multiple controls

id="namelabel"

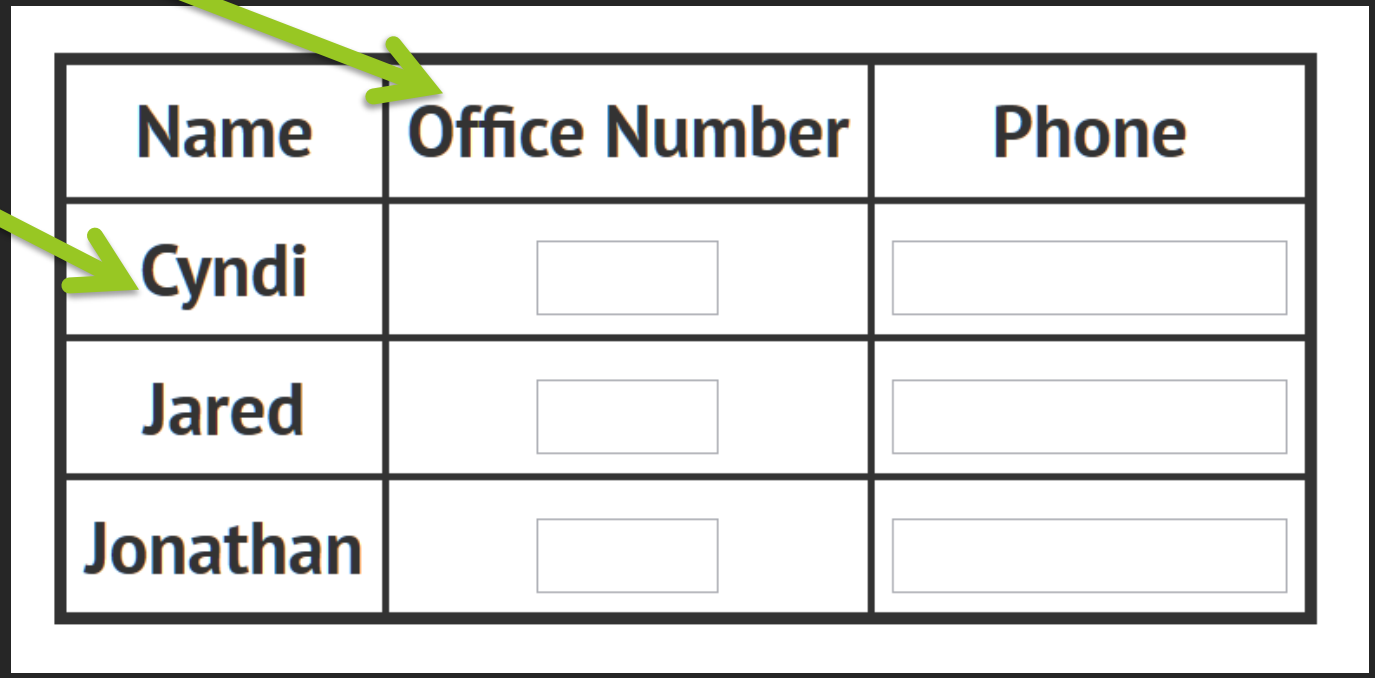



```
<input type="text" name="name1"
aria-labelledby="namelabel">
```

Multiple labels for multiple controls

id="officenum"

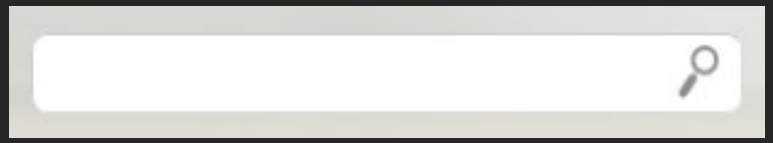
id="cyndi"



Name	Office Number	Phone
Cyndi		
Jared		
Jonathan		

```
<input type="text" name="office1"
aria-labelledby="cyndi officenum">
```

aria-label



```
<input aria-label="Search Terms"…>
```

ARIA *labels* override default accessible *names*



Accessible Name

```
<label for="fname">First Name:</label>
<input type="text" id="fname"…
aria-label="Your Name">
```

…and not just in forms




```
<a… aria-label="Change shipping
method">Standard Domestic - $4.95 USD</a>
```

Also applies to alternative text, button text, etc.

cannot have an accessible name

Download the Employment Application 


```
<a href="app.pdf">Download the Employment Application
<span class="PDFlink" aria-label="PDF"></span>
</a>
```

Neither can , <p>, <i>, etc.

Three possible solutions

Download the Employment Application 

1. `<a class="PDFlink">Download the Employment Application<span class="sr-only"> (PDF)</span></a>`
2. `<a class="PDFlink">Download the Employment Application<span class="PDFlink" role="img" aria-label="(PDF)"></a>`
3. `<a class="PDFlink" aria-label="Download the Employment Application (PDF)">Download the Employment Application</a>`

CSS generated content



```
<i class="pdf"></i>
```

```
.pdf::after {
  font-family: FontAwesome;
  content: "\f1c1";
}
```

CSS generated content will be read in modern screen readers that understand the generated content.

CSS generated content



```
<i class="pdf" role="img" aria-label="PDF"></i>
```

or

```
<button aria-label="PDF"><i class="pdf"></i></button>
```

Scalable Vector Graphics (SVG)

1. `<img src="chart.svg" alt="Sales increased 10% from 2010 to 2015">`

2. `<svg role="img" aria-labelledby="title">`
   `<title id="title">Sales increased 10% from 2010 to 2015</title>`
   `…</svg>`

3. `<svg role="img" aria-label="Sales increased 10% from 2010 to 2015">`
   `…</svg>`

Use `role="presentation"` and `aria-hidden="true"` if the SVG is decorative (or `alt=""` for <img>).

aria-describedby

Username:

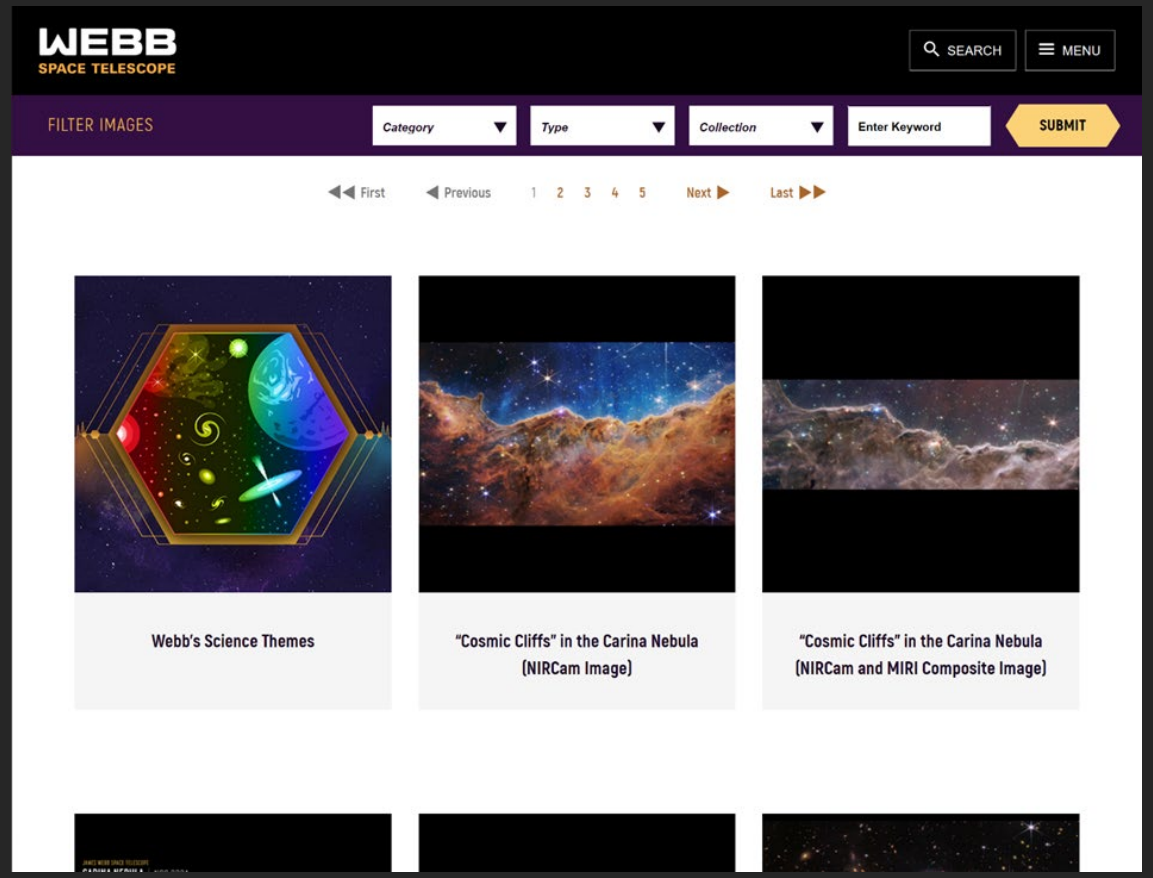
Must be 8-15 characters

```
<label for="user">Username:</label>
<input type="text" id="user"
aria-describedby="usernamedetails">
<p id="usernamedetails">Must be 8-15 characters</p>
```

`title` attribute as a description

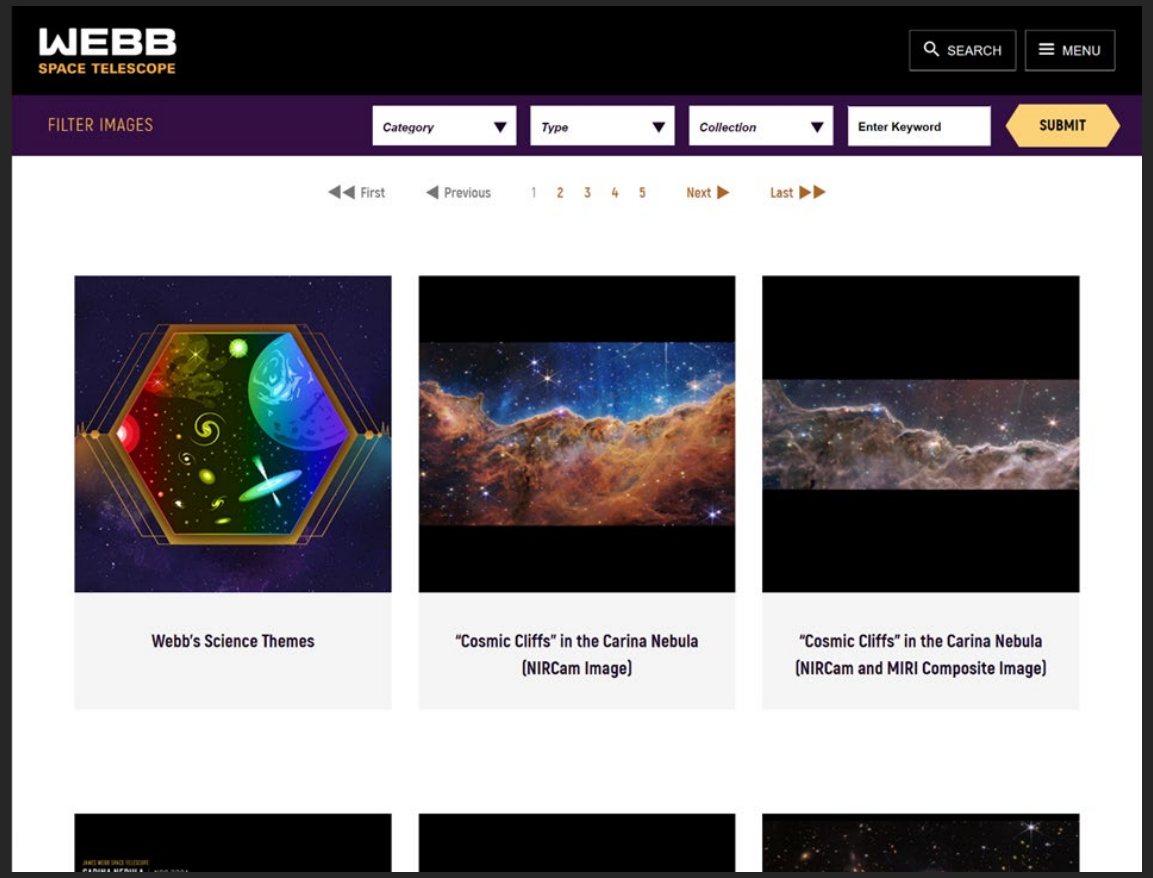
- Advisory information only
- Not accessible to keyboard users, touch screen users, etc.
- Accessible name if there are no other options. E.g.,
  - `<input>` with no `<label>` or ARIA label
  - `<img>` with no `alt` or ARIA label
  - Link with no text or ARIA label
- Otherwise, it should be read as a description

Caution with redundant title text




```
title="Read more about Image of bright eight-pointed stars scattered across a dense orange-brown undulating cloudscape below a deep blue starscape."
```

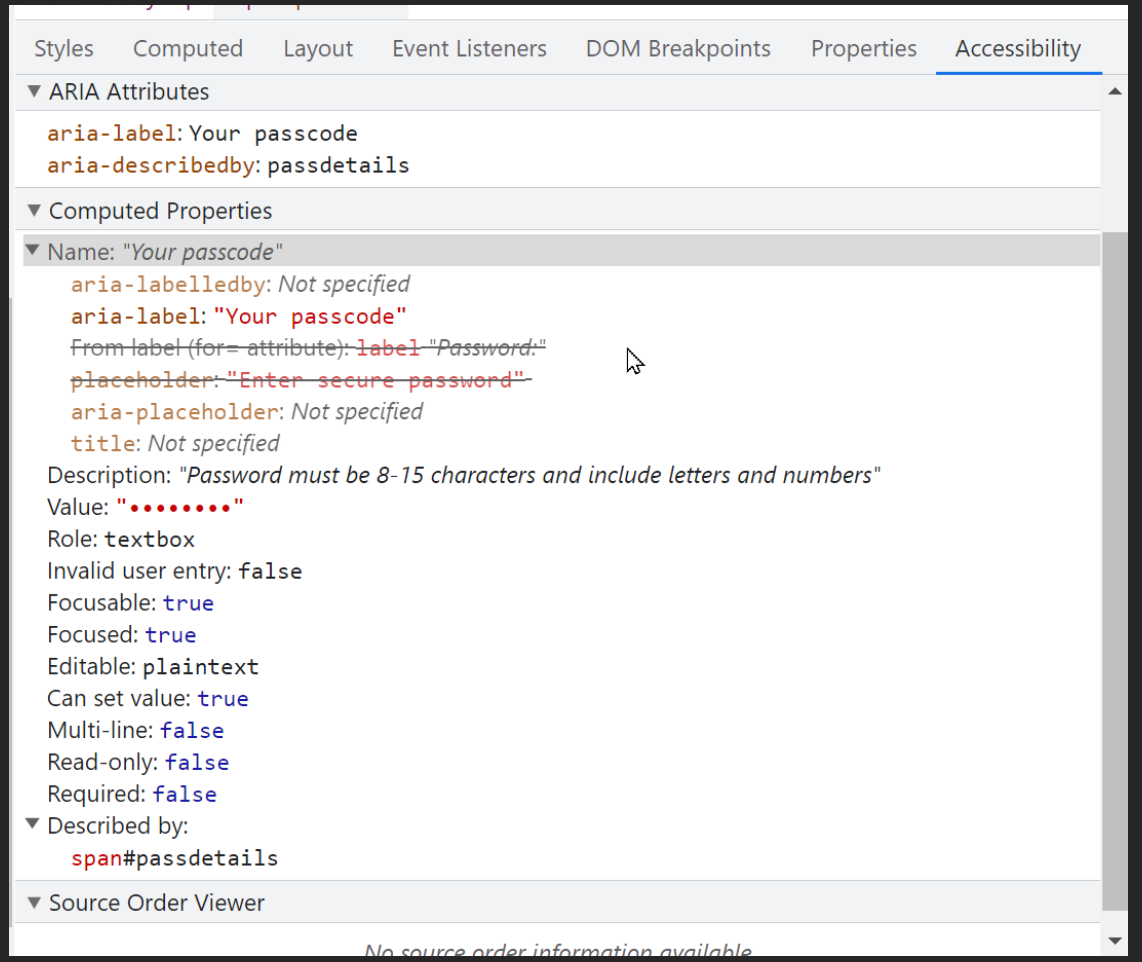
The result for most screen readers



"Image of bright eight-pointed stars scattered across a dense orange-brown undulating cloudscape below a deep blue starscape. "Cosmic Cliffs" in the Carina Nebula (NIRCam Image) Link Read more about Image of bright eight-pointed stars scattered across a dense orange-brown undulating cloudscape below a deep blue starscape."

Evaluate accessible name and description with Chrome DevTools

1. Use “Inspect Mode Tooltip” (top-left corner) to inspect the form field.
2. Open accessibility tab
3. Review “Accessibility Tree” and “ARIA Attributes”
4. Using “Computed Properties”, find the accessible name and description



ARIA and HTML labels and descriptions

- Text in `aria-label` is ~~not~~ automatically translated (e.g., Google Translate)
- Are read as a stream of text. Users cannot easily navigate or explore the referenced content
  - Semantics (lists, links, etc.) are removed from referenced content.

Username:

Must:

- *be at least 14 characters in length*
- *contain at least 4 vowels*
- *contain one medieval rune of alien origin*

Username:

Must meet these requirements

Hidden ARIA and HTML labels and descriptions

- `<label>` hidden with CSS `display:none` or `hidden` **will not** be read by screen readers.
- Elements hidden with `display:none` or `hidden`, but referenced by `aria-labelledby` or `aria-describedby` **WILL** be read.

```
<label for="user">Username:</label>
<input type="text" id="user"
aria-describedby="usernameerror">
<p id="usernameerror"
style="display:none">This username is already
taken. Please try again.</p>
```

Form Errors

Help users avoid errors when possible

- Provide clear labels or instructions
- Important changes (e.g., Financial) are reversible, checked, or confirmed
- Don’t require re-entering information (WCAG 2.2)
- Don’t collect unnecessary information
- Be flexible in formatting (e.g., phone numbers)

aria-required

Username:* 

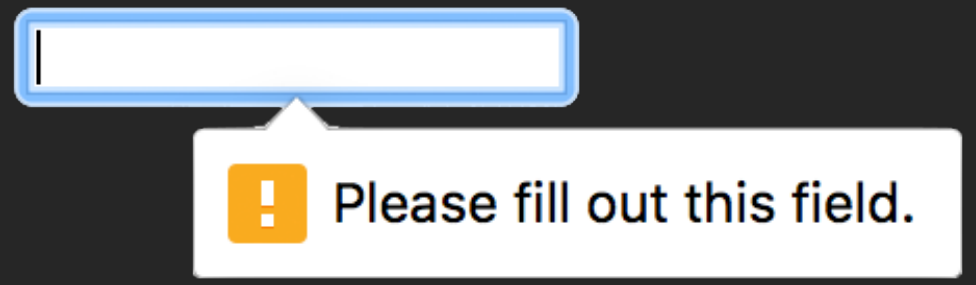
```
<label for="username">Username:</label>*
<input type="text" id="username"
aria-required="true">
```

A screen reader indicates the field is required…

and that's all.

HTML required

Username:*




```
<label for="username">Username:</label>*
<input type="text" id="username" required>
```

Semantics are the same, but validation messaging is provided

Pay with debit or credit card

We don't share your financial details with the merchant.

United States



Card number

Expires

MM YY

CSC

3 digits

First name

Last name

Email

Mobile

Phone number

Billing address

Street address

Apt., ste., bldg., etc. (optional)

City

State

ZIP

Help users correct mistakes if they occur

- Identify the error
  - There is an error
  - Here is the error
  - This is the error
- Suggest a correction
  - E.g., “Password must include letters and numbers.”

aria-invalid

Password: 

```
<label for="password">Password:</label>
<input type="password" id="password"
aria-invalid="true"…>
```

A screen reader indicates the field is invalid or broken…

and that’s all.

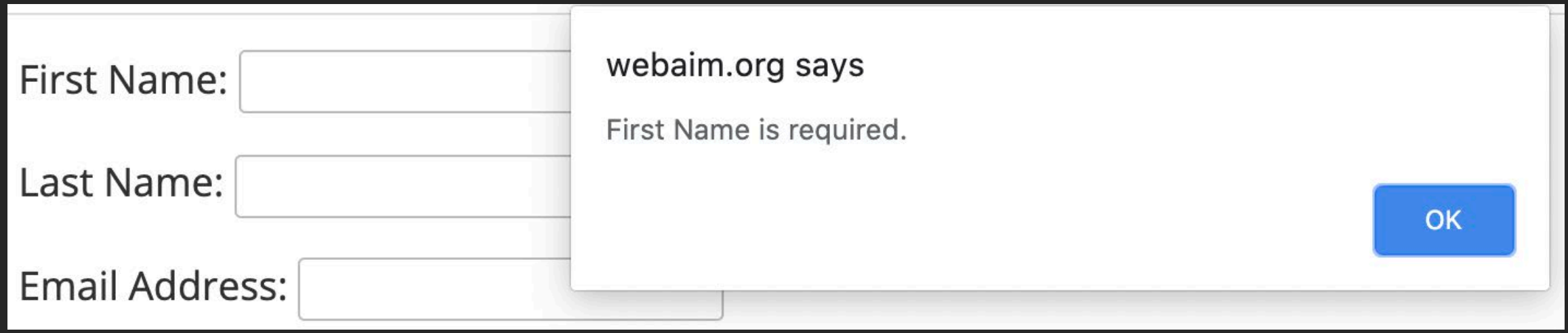
Use ARIA attributes to control styling:

```
[aria-invalid=true] {
  border: 2px solid red;
  background-image: url("error.png");
}
```

Types of error identification

- Alert, then focus
- Errors at the top
- Inline errors
- Status messages

Alert, then focus

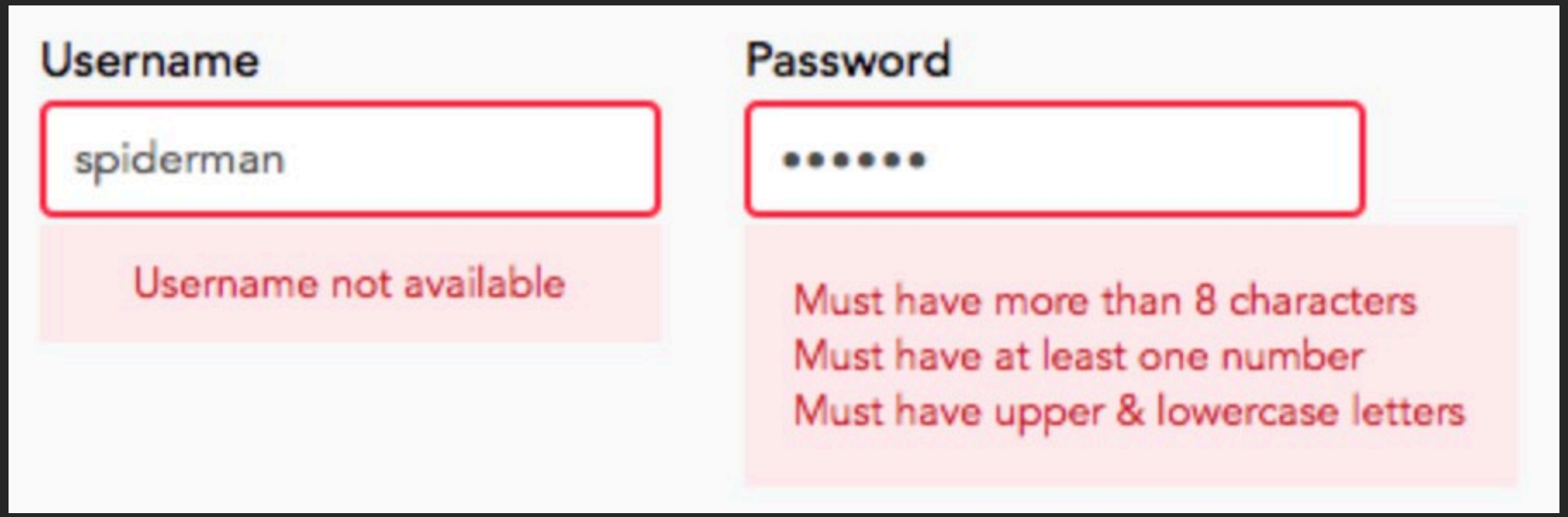


Errors at the top

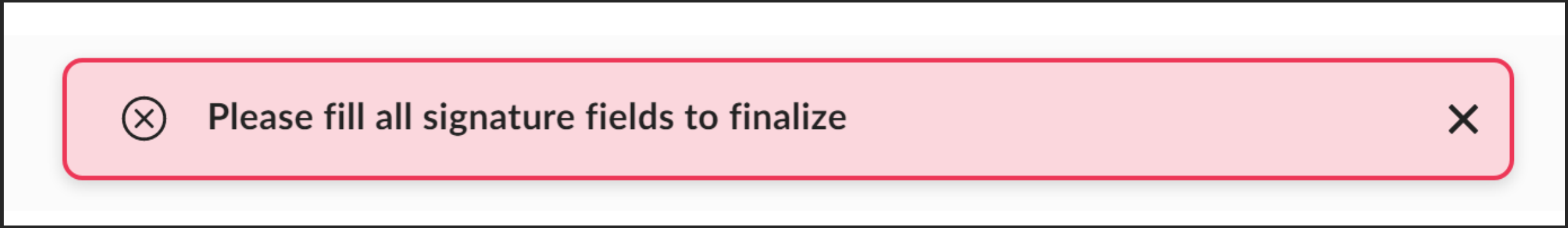
There was a problem with the form.

- Please enter Your Name.
- Please enter Your E-mail Address.

Inline Errors



Status Messages



SC 4.1.3 Status Messages (AA)

- ARIA live regions
  - `role="alert"/aria-live="assertive"` – Read now
  - `aria-live="polite"` – Read at a pause
- Don’t set focus to status messages that disappear automatically.
- Multiple messages? Queue messages in a hidden messaging element.

ARIA Applied
(and Misapplied)

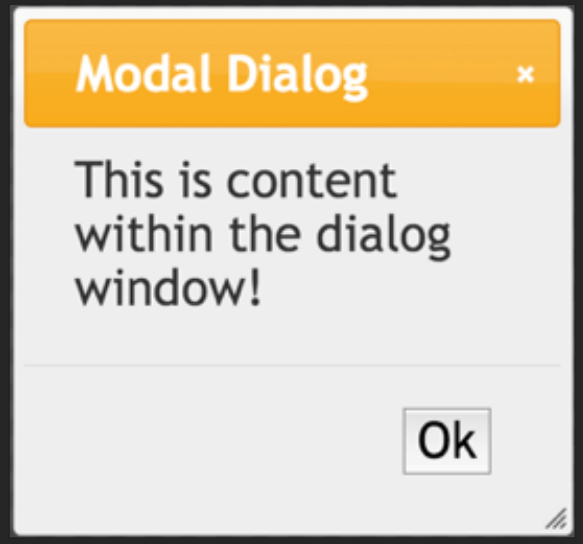
`role="dialog"`
w/ `aria-labelledby`

`tabindex="-1"` then set focus to dialog (or focus a control or element inside the dialog)

Maintains keyboard focus if modal

Button with `aria-haspopup="dialog"`





Returns focus when dismissed

Closes with ESC key

Dialog focus management

- Current:
  - Background: `tabindex="-1"` + `aria-hidden="true"`
  - Dialog: Scripted focus detection and re-focus + `aria-modal`
- Future:

```
<div inert>
   page content
</div>
<dialog modal>
   dialog content
</dialog>
```

Differentiating regions/landmarks

```
<div role="navigation" aria-label="Main navigation">

    <nav aria-label="Page navigation">
```

Other regions

```
<div role="region" aria-labelledby="filterheading">
  <h2 id="filterheading">Filters</h2>
  …
</div>
```

Generic regions must have an accessible name.

More ARIA

```
<button aria-expanded="false">Details</button>
```

webaim.org/presentations/2023/examples/disclosure.htm

```
<button aria-pressed="true">Toggle Highlights</button>
```

webaim.org/presentations/2023/examples/ariapressed.htm

```
<a aria-current="page | step | location | date | time | true">
```

Windows screen reader modes

- Reading / Virtual Cursor / Document
- Forms / Application

The current mode determines whether the screen reader or the browser handles most keyboard commands

Some ARIA roles (tree, slider, grid, tabpanel, menu, etc.) trigger forms/application mode.

You must ensure the user is aware and that the proper keyboard interactions are implemented.

Test with AND without a Windows screen reader.

ARIA

With great power comes great responsibility!

Thank You!

Services

- Training
  - Strategica11y
- Evaluation
- Consultation
  - System-wide coordination

Resources

- E-mail discussion list
- Monthly newsletter
- Articles and resources
- WebAIM conference

WebAIM
web accessibility in mind

