endava

Website Accessibility Webinar

Today we will be discussing five topics

1. Accessibility Overview
2. Introduction to Assistive Technology (AT)
3. Web Content Accessibility Guidelines (WCAG) 2.1
4. WebAIM's WCAG 2 Checklist
5. Implementation Details

1

THE FUNDAMENTALS:

Accessibility Overview

IMPROVING ACCESSIBILITY

ADA

WHAT IS ACCESSIBILITY?

PRACTICES TO BUILD SOFTWARE SYSTEMS WHICH ARE USABLE AND UNDERSTANDABLE BY USERS WITH DISABILITIES

VISUAL IMPAIRMENTS

EXAMPLES

- BLIND USERS
- LOW VISION USERS
- COLOR BLIND USERS

ASSISTIVE TECHNOLOGY

- SCREEN READERS
- BRAILLE READERS
- ZOOM
- HIGH CONTRAST MODE

MOTOR IMPAIRMENTS

EXAMPLES

- PHYSICALLY PARALYZED
- CARPAL TUNNEL
- BROKEN ARM (TEMPORARY)

ASSISTIVE TECHNOLOGY

- VOICE CONTROL
- EYE-TRACKING DEVICE
- SWITCH DEVICE

HEARING IMPAIRMENTS

EXAMPLES

- DEAF
- HARD-OF-HEARING

ASSISTIVE TECHNOLOGY

- CLOSED CAPTIONS
- DESIGN CONSIDERATIONS

COGNITIVE IMPAIRMENTS

EXAMPLES

- EPILEPSY
- DYSLEXIA
- AUTISM

ASSISTIVE TECHNOLOGY

- ZOOM
- MINIMAL DESIGN

WHY IS ACCESSIBILITY IMPORTANT?

RIGHT THING TO DO

EVERYONE DESERVES AN EQUAL OPPORTUNITY -
IMPROVE BRAND IMAGE AND SHOW CULTURE -

LEGAL LIABILITY

- TITLE III OF ADA APPLIES TO MANY BUSINESSES
- ROBLES V.S. DOMINOS PIZZA CASE
- WCAG 2.0 GUIDELINES ESTABLISH COMMON STANDARDS

INCREASE YOUR CUSTOMER BASE

18.7% OF US POPULATION HAVE A DISABILITY -
12.6% HAVE A SEVERE DISABILITY -

BEST PRACTICES OVERLAP

- UI & ACCESSIBILITY BEST PRACTICES OVERLAP
- STANDARDS CAN IMPROVE SEO, MOBILE X, UI
- INCREASE MAINTAINABILITY AND IMPROVE REVIEWS

2

TOOLS OF THE TRADE:

Introduction to Assistive Technology (AT)

INTRODUCTION TO ASSISTIVE TECHNOLOGY (AT)

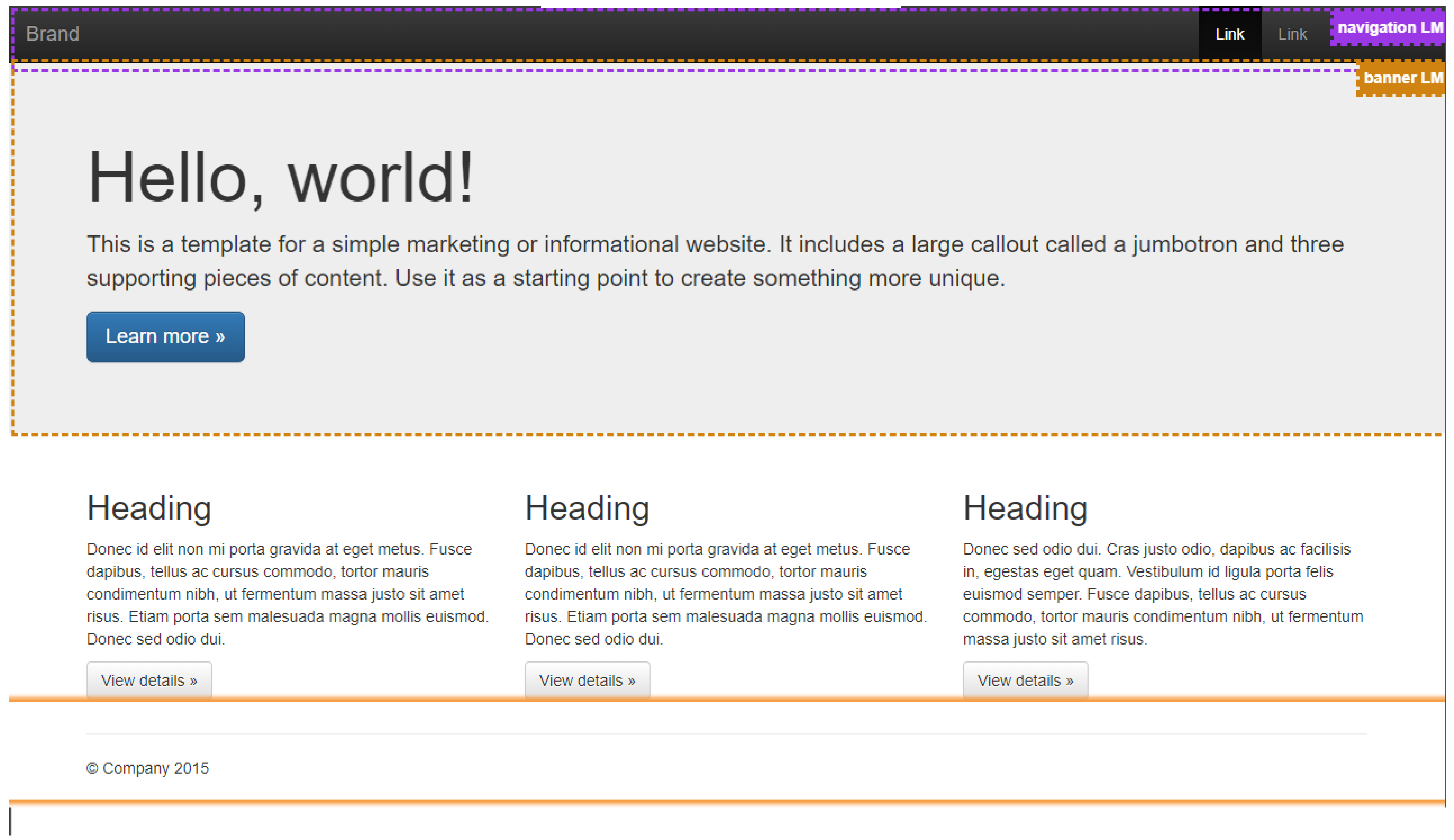
Screen Readers

Screen readers are a form of assistive technology (AT) used by blind or visually impaired users that render content as speech or Braille output.

They provide a rich set of keyboard navigation commands to efficiently traverse page or screen hierarchy.

Supporting screen readers will cover approximately 80% of WCAG 2.1 requirements. These include text alternatives, keyboard navigation, and semantic structure.

Common screen readers include ChromeVox Classic Extension (Chrome), VoiceOver (iOS), and TalkBack (Android).



Hello, world!

This is a template for a simple marketing or informational website. It includes a large callout called a jumbotron and three supporting pieces of content. Use it as a starting point to create something more unique.

Learn more »

Heading

Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

View details »

Heading

Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui.

View details »

Heading

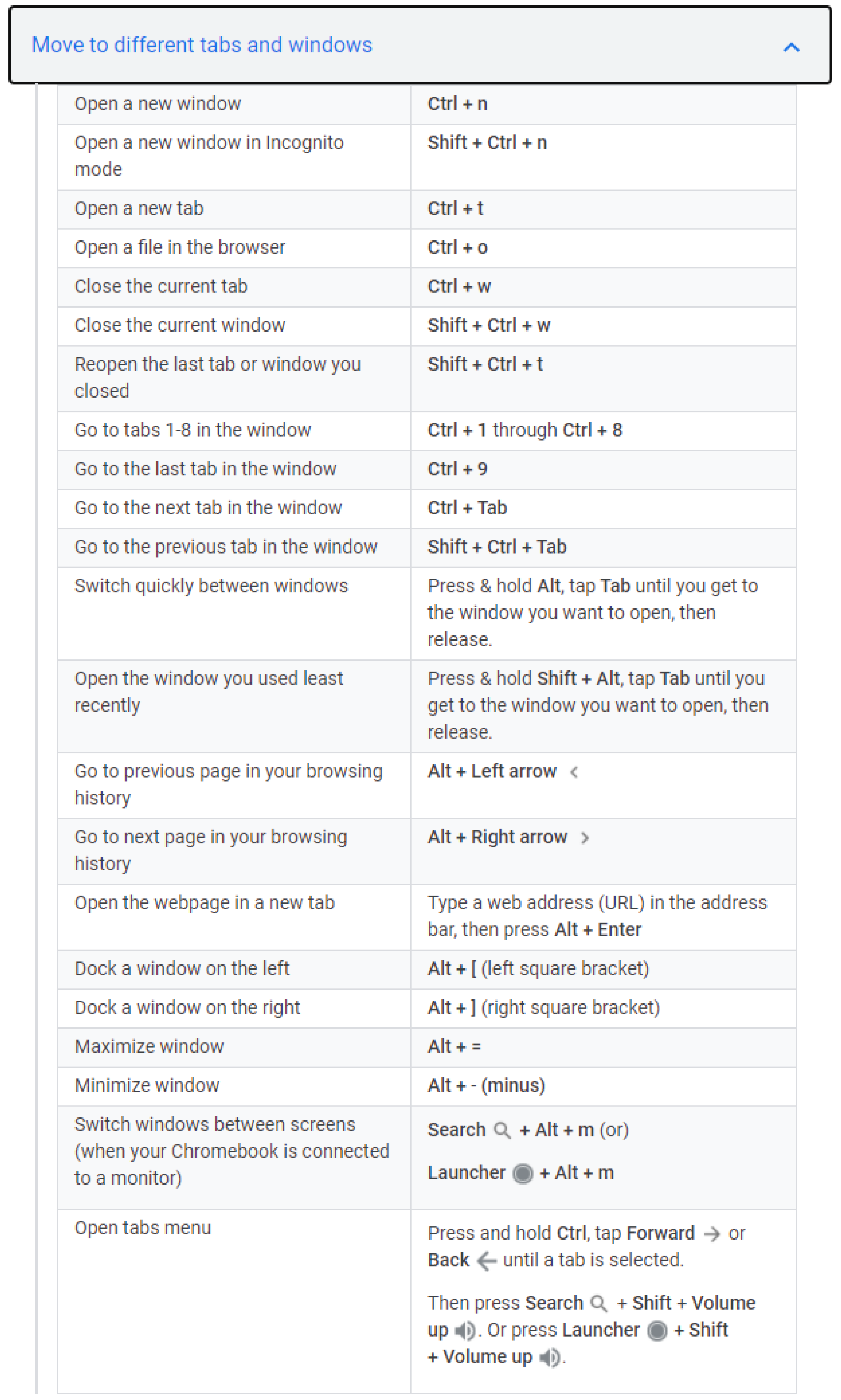
Donec sed odio dui. Cras justo odio, dapibus ac facilisis in, egestas eget quam. Vestibulum id ligula porta felis euismod semper. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.

View details »

© Company 2015

ACCESSIBILITY OVERVIEW:

Screen Reader Navigation and Keyboard Traps



Move to different tabs and windows

Open a new window	**Ctrl + n**
Open a new window in Incognito mode	**Shift + Ctrl + n**
Open a new tab	**Ctrl + t**
Open a file in the browser	**Ctrl + o**
Close the current tab	**Ctrl + w**
Close the current window	**Shift + Ctrl + w**
Reopen the last tab or window you closed	**Shift + Ctrl + t**
Go to tabs 1-8 in the window	**Ctrl + 1** through **Ctrl + 8**
Go to the last tab in the window	**Ctrl + 9**
Go to the next tab in the window	**Ctrl + Tab**
Go to the previous tab in the window	**Shift + Ctrl + Tab**
Switch quickly between windows	Press & hold **Alt**, tap **Tab** until you get to the window you want to open, then release.
Open the window you used least recently	Press & hold **Shift + Alt**, tap **Tab** until you get to the window you want to open, then release.
Go to previous page in your browsing history	**Alt + Left arrow** ‹
Go to next page in your browsing history	**Alt + Right arrow** ›
Open the webpage in a new tab	Type a web address (URL) in the address bar, then press **Alt + Enter**
Dock a window on the left	**Alt + [** (left square bracket)
Dock a window on the right	**Alt +]** (right square bracket)
Maximize window	**Alt + =**
Minimize window	**Alt + - (minus)**
Switch windows between screens (when your Chromebook is connected to a monitor)	**Search** 🔍 **+ Alt + m** (or) **Launcher** ● **+ Alt + m**
Open tabs menu	Press and hold **Ctrl**, tap **Forward** → or **Back** ← until a tab is selected. Then press **Search** 🔍 + **Shift** + **Volume up** 🔊. Or press **Launcher** ● + **Shift** + **Volume up** 🔊.

Technology, P. (2021) *Using ChromeVox on Your Chomebook: Video Tutorials, Perkinselearning.org*. Available at: https://www.perkinselearning.org/technology/posts/using-chromevox-your-chomebook-video-tutorials (Accessed: 29 Ma
2021).

Alternative Input Devices

The typical software system implemented for keyboard, mouse and/or touch-screen input device. Disable users use a wide range of alternative input devices such as:

- Keyboard
- Braille board
- Bluetooth Keyboard
- Switch device
- Cheek switch device
- Head pointers
- Speech input software

Braille board

Cheek switch device

Display Augmentation Accessibility Tools

There are a significant number of tools used to augment the display for improved accessibility such as:

- Screen magnification / zoom
- Font size settings
- High contrast mode

3

THE COMPLIANCE GUIDELINES:

Web Content Accessibility Guidelines (WCAG) 2.1

ACCESSIBILITY OVERVIEW:

Web Content Accessibility Guidelines (WCAG) 2.1

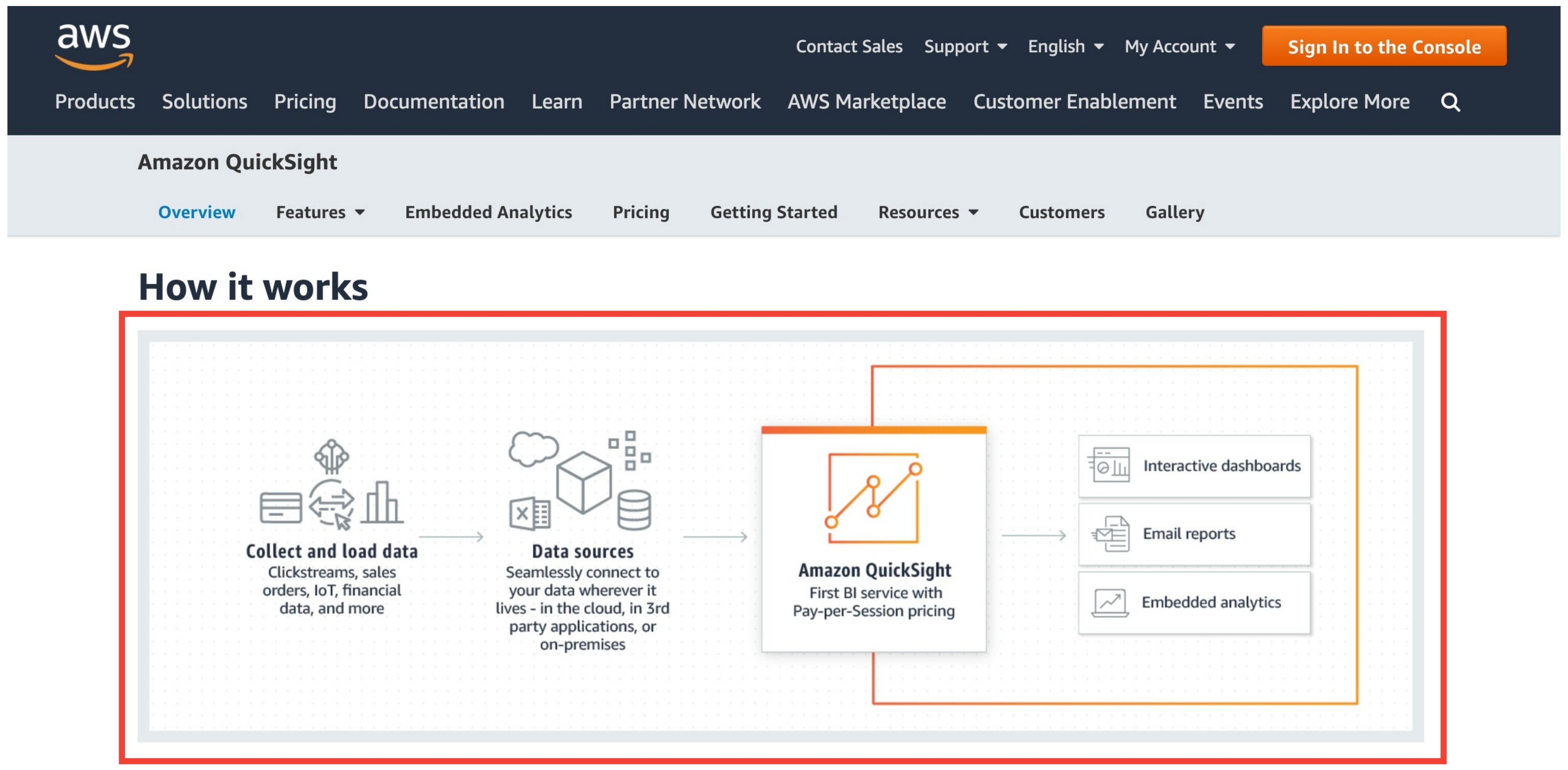
- Set of guidelines put together by W3C - https://www.w3.org/TR/WCAG21/ - (61 in all)
- Not prescriptive and only describe functionality needed
- Related documents explain success criteria for Web. None available for Native Apps
- WCAG includes four sections often called by the mnemonic **POUR**
  - **Perceivable**: Information and user interface components must be presentable to users in ways they can perceive.
  - **Operable**: User interface components and navigation must be operable.
  - **Understandable**: Information and the operation of user interface must be understandable.
  - **Robust**: Content must be robust enough that it can be interpreted by by a wide variety of user agents, including assistive technologies.

ACCESSIBILITY OVERVIEW:

Perceivable

Can users sense and understand the content?

- General
  - Provide text alternatives for non-text content
  - Provide captions, controls, etc. for multimedia
  - Provide semantic markup to convey meaning
  - Do not convey meaning through color
  - Contrast ratio considerations
  - Text size considerations
- Native App
  - Screen size consideration
  - Screen magnifiers
  - Dynamic type sizing
  - Focus controls



How it works

Collect and load data
Clickstreams, sales orders, IoT, financial data, and more

Data sources
Seamlessly connect to your data wherever it lives - in the cloud, in 3rd party applications, or on-premises

Amazon QuickSight
First BI service with Pay-per-Session pricing

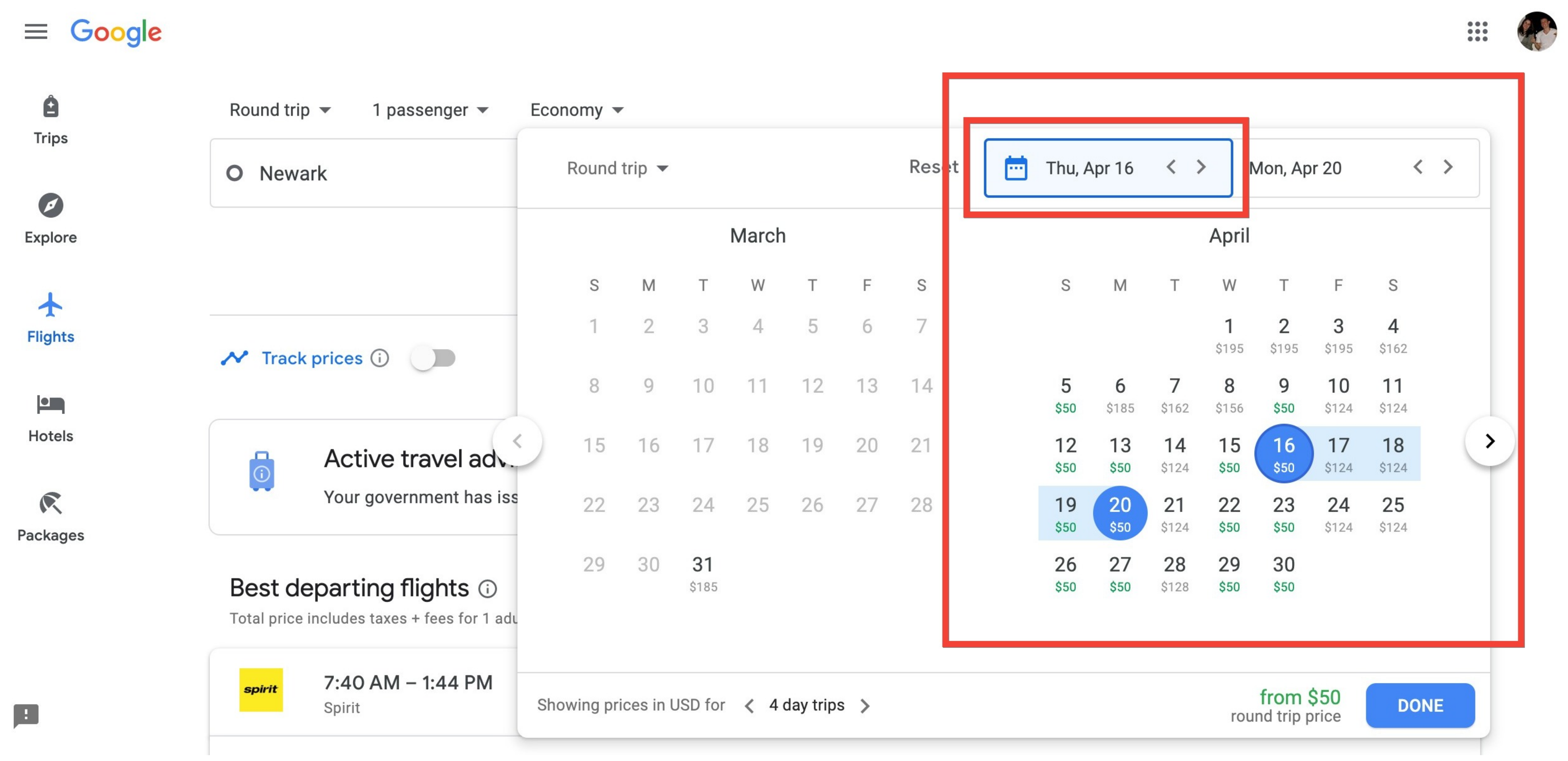
This image shows an image of text being used in violation of WCAG 1.4.9.

ACCESSIBILITY OVERVIEW:

Operable

Can users navigate the UI and use the functionality?

- General
  - All functionality available from a keyboard
  - Give sufficient time and control of timed events
  - Do not design content that may cause seizures
  - Use semantic markup to reinforce meaning of content
- Native App
  - Implement input focus (keyboard)
  - Implement accessibility focus (TalkBack and VoiceOver)
  - Minimize advanced gestures
  - Touch target size considerations



Round trip

Thu, Apr 16 — Mon, Apr 20

March

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31 $185				

April

S	M	T	W	T	F	S
			1 $195	2 $195	3 $195	4 $162
5 $50	6 $185	7 $162	8 $156	9 $50	10 $124	11 $124
12 $50	13 $50	14 $124	15 $50	16 $50	17 $124	18 $124
19 $50	20 $50	21 $124	22 $50	23 $50	24 $124	25 $124
26 $50	27 $50	28 $128	29 $50	30 $50		

7:40 AM – 1:44 PM Spirit

Showing prices in USD for 4 day trips

from $50 round trip price

This image shows a date picker that still functions properly for keyboards as an HTML input field.

ACCESSIBILITY OVERVIEW:

Understandable

Can users understand the UI and is it consistent?

- General
  - Define the language of the page
  - Provide a predictable UI
  - Help minimize mistakes (error validation, labels, etc.)
- Native App
  - Support multiple screen orientations
  - Provide a consistent layout

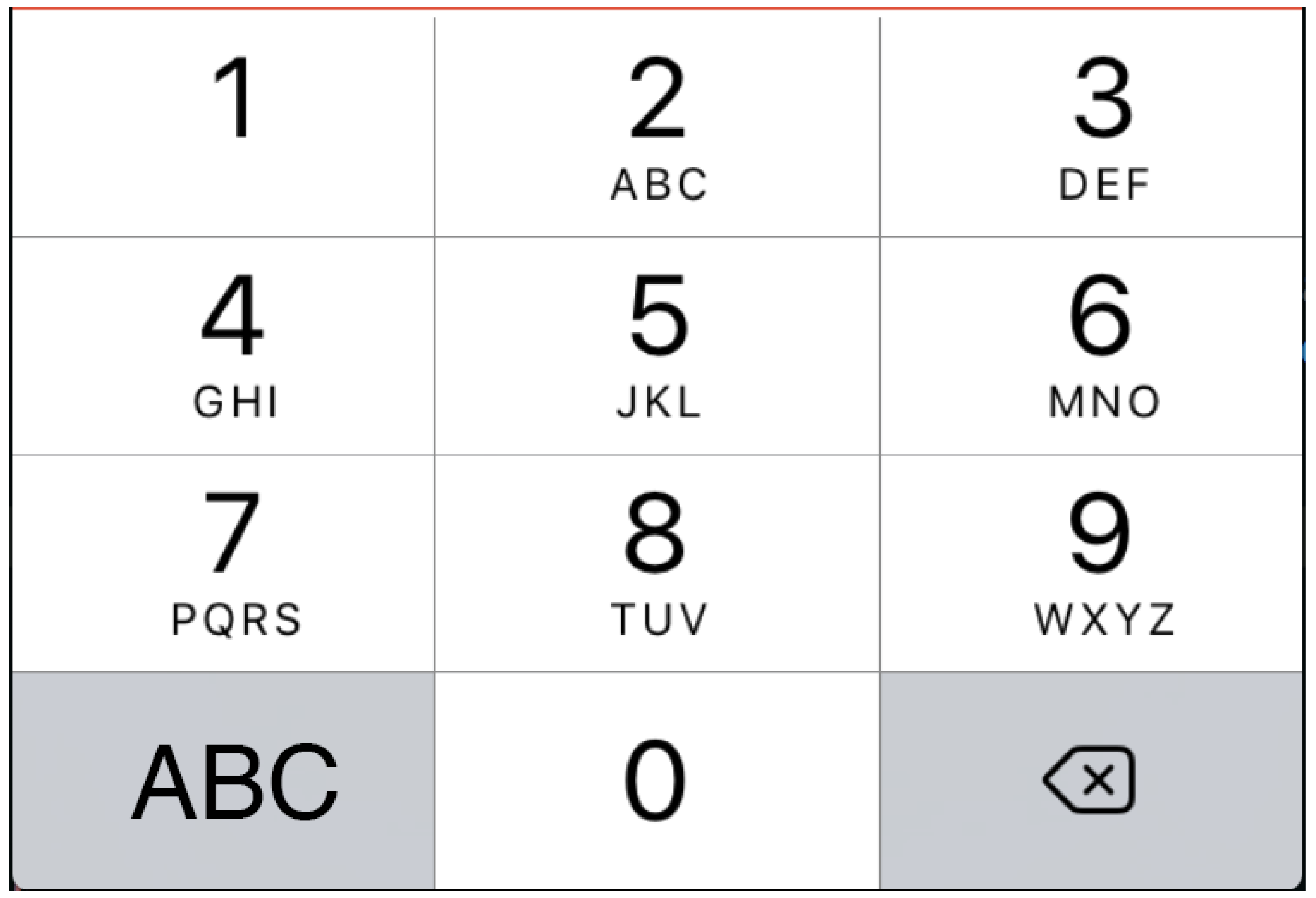
This image shows Eric Legrand with a fixed device orientation.

ACCESSIBILITY OVERVIEW:

Robust

Can assistive technology interpret the UI?

- General
  - Utilize platform features to increase accessibility (WAI-ARIA, etc.)
  - Use common platform UI elements (<form>, android.widget.Button, etc.)
  - Announce UI status changes (Loading, etc.)
- Native App
  - Operate in same way as system
  - Keyboard type



This image shows a numeric keypad on iOS designed for number and PIN entry.

4

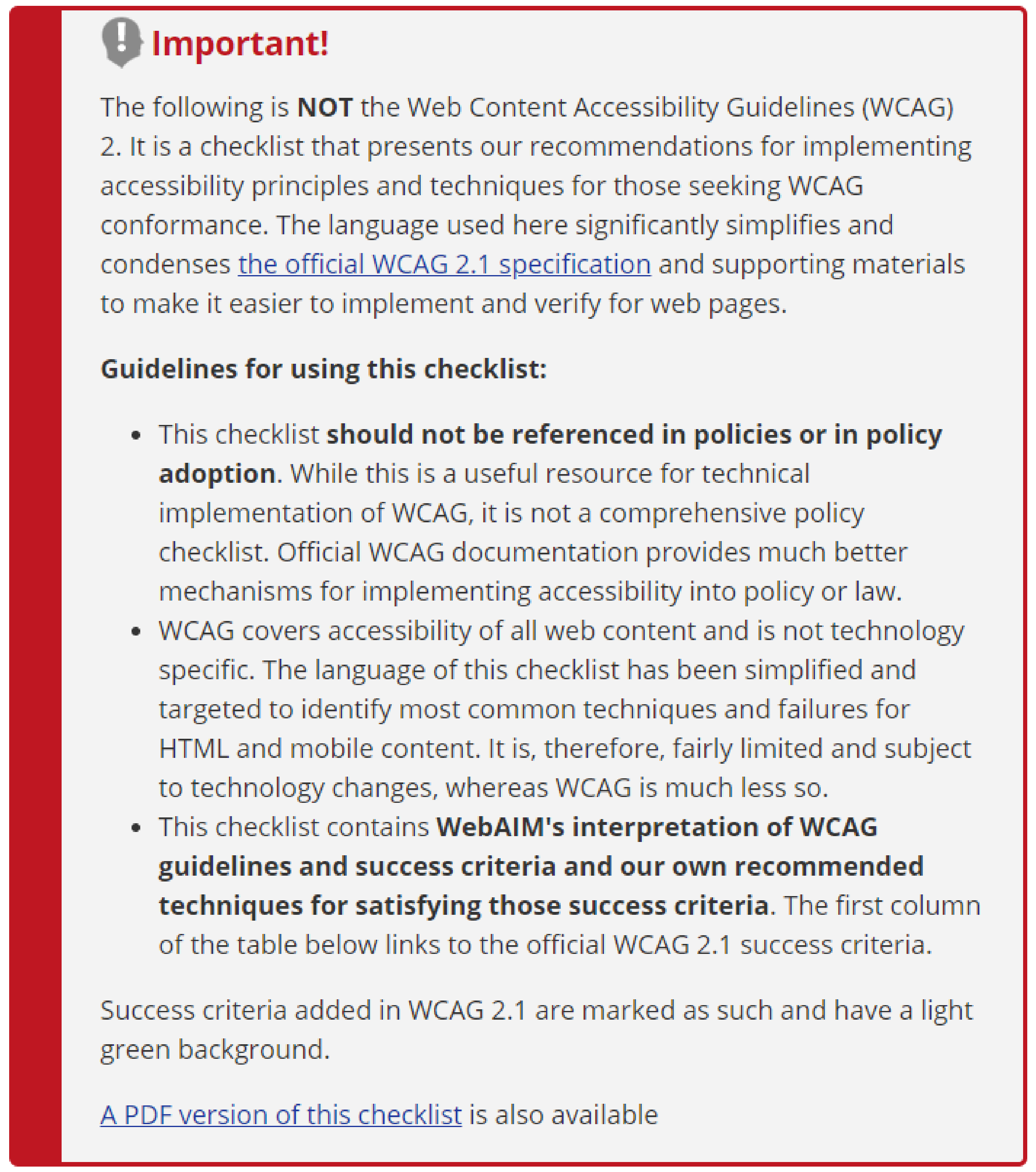
WCAG 2 CHECKLIST:

WebAIM's WCAG 2 Checklist

ACCESSIBILITY OVERVIEW:

WebAIM's WCAG 2 Checklist

- Checklist that presents WebAIM's recommendations for implementing accessibility principles and techniques - https://webaim.org/standards/wcag/checklist



Important!

The following is **NOT** the Web Content Accessibility Guidelines (WCAG) 2. It is a checklist that presents our recommendations for implementing accessibility principles and techniques for those seeking WCAG conformance. The language used here significantly simplifies and condenses the official WCAG 2.1 specification and supporting materials to make it easier to implement and verify for web pages.

Guidelines for using this checklist:

- This checklist **should not be referenced in policies or in policy adoption**. While this is a useful resource for technical implementation of WCAG, it is not a comprehensive policy checklist. Official WCAG documentation provides much better mechanisms for implementing accessibility into policy or law.
- WCAG covers accessibility of all web content and is not technology specific. The language of this checklist has been simplified and targeted to identify most common techniques and failures for HTML and mobile content. It is, therefore, fairly limited and subject to technology changes, whereas WCAG is much less so.
- This checklist contains **WebAIM's interpretation of WCAG guidelines and success criteria and our own recommended techniques for satisfying those success criteria**. The first column of the table below links to the official WCAG 2.1 success criteria.

Success criteria added in WCAG 2.1 are marked as such and have a light green background.

A PDF version of this checklist is also available

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 1.1

Text Alternatives: Provide text alternatives for any non-text content

Success Criteria	WebAIM's Recommendations
1.1.1 Non-text Content (Level A)	•Images, form image buttons, and image map hot spots have appropriate, equivalent alternative text. •Images that do not convey content, are decorative, or contain content that is already conveyed in text are given empty alternative text (alt="") or implemented as CSS backgrounds. All linked images have descriptive alternative text. •Equivalent alternatives to complex images are provided in context or on a separate linked page. •Form buttons have a descriptive value. •Form inputs have associated text labels. •Embedded multimedia is identified via accessible text. •Frames and iframes are appropriately titled.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 1.2

Time-based Media: Provide alternatives for time-based media

Success Criteria	WebAIM's Recommendations
1.2.1 Prerecorded Audio-only and Video-only (Level A)	•A transcript of relevant content is provided for non-live audio-only (audio podcasts, MP3 files, etc.). •A transcript or audio description of relevant content is provided for non-live video-only, unless the video is decorative.
1.2.2 Captions (Prerecorded) (Level A)	•Synchronized captions are provided for non-live video (YouTube videos, etc.).
1.2.3 Audio Description or Media Alternative (Prerecorded) (Level A)	•A transcript or audio description is provided for non-live video. NOTE: Only required if there is relevant visual content that is not presented in the audio.
1.2.4 Captions (Live) (Level AA)	•Synchronized captions are provided for live media that contains audio (audio-only broadcasts, web casts, video conferences, etc.)
1.2.5 Audio Description (Prerecorded) (Level AA)	•Audio descriptions are provided for non-live video. NOTE: Only required if there is relevant visual content that is not presented in the audio. •While not required at Level AA, for optimal accessibility WebAIM recommends transcripts in addition to audio descriptions.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 1.3

Adaptable: Create content that can be presented in different ways (for example simpler layout) without losing information or structure

Success Criteria	WebAIM's Recommendations
1.3.1 Info and Relationships (Level A)	•Semantic markup is used to designate headings (<h1>), regions/landmarks, lists (<ul>, <ol>, and <dl>), emphasized or special text (<strong>, <code>, <abbr>, <blockquote>, for example), etc. Semantic markup is used appropriately. •Tables are used for tabular data and data cells are associated with their headers. Data table captions, if present, are associated to data tables. •Text labels are associated with form input elements. Related form elements are grouped with fieldset/legend. ARIA labelling may be used when standard HTML is insufficient.
1.3.2 Meaningful Sequence (Level A)	•The reading and navigation order (determined by code order) is logical and intuitive.
1.3.3 Sensory Characteristics (Level A)	•Instructions do not rely upon shape, size, or visual location (e.g., "Click the square icon to continue" or "Instructions are in the right-hand column"). •Instructions do not rely upon sound (e.g., "A beeping sound indicates you may continue.").

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 1.4

Distinguishable: Make it easier for users to see and hear content including separating foreground from background

Success Criteria	WebAIM's Recommendations
1.4.1 Use of Color (Level A)	•Color is not used as the sole method of conveying content or distinguishing visual elements. •Color alone is not used to distinguish links from surrounding text unless the contrast ratio between the link and the surrounding text is at least 3:1 and an additional distinction (e.g., it becomes underlined) is provided when the link is hovered over and receives focus.
1.4.2 Audio Control (Level A)	•A mechanism is provided to stop, pause, mute, or adjust volume for audio that automatically plays on a page for more than 3 seconds.
1.4.3 Contrast (Minimum) (Level AA)	•Text and images of text have a contrast ratio of at least 4.5:1. •Large text - at least 18 point (typically 24px) or 14 point (typically 18.66px) and bold - has a contrast ratio of at least 3:1.
1.4.4 Resize text (Level AA)	•The page is readable and functional when the page is zoomed to 200%. NOTE: 1.4.10 (below) introduces a much higher requirement for zoomed content.
1.4.5 Images of Text (Level AA)	•If the same visual presentation can be made using text alone, an image is not used to present that text.
1.4.10 Reflow (WCAG 2.1 Level AA)	•No loss of content or functionality occurs and horizontal scrolling is avoided when content is presented at a width of 320 pixels. • This requires responsive design for most web sites. This is best tested by setting the browser window to 1280 pixels wide and then zooming the page content to 400%. •Content that requires horizontal scrolling, such as data tables, complex images (such as maps and charts), toolbars, etc. are exempted.
1.4.11 Non-text Contrast (WCAG 2.1 Level AA)	•A contrast ratio of at least 3:1 is present for differentiating graphical objects (such as icons and components of charts or graphs) and author-customized interface components (such as buttons, form controls, and focus indicators/outlines). •At least 3:1 contrast must be provided in the various states (focus, hover, active, etc.) of author-customized interactive components.
1.4.12 Text Spacing (WCAG 2.1 Level AA)	•No loss of content or functionality occurs when the user adapts paragraph spacing to 2 times the font size, text line height/spacing to 1.5 times the font size, word spacing to .16 times the font size, and letter spacing to .12 times the font size. •This is best supported by avoiding pixel height definitions for elements that contain text.
1.4.13 Content on Hover or Focus (WCAG 2.1 Level AA)	•When additional content is presented on hover or keyboard focus: • The newly revealed content can be dismissed (generally via the Esc key) without moving the pointer or keyboard focus, unless the content presents an input error or does not obscure or interfere with other page content. • The pointer can be moved to the new content without the content disappearing. • The new content must remain visible until the pointer or keyboard focus is moved away from the triggering control, the new content is dismissed, or the new content is no longer relevant.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 2.1

Keyboard Accessible: Make all functionality available from a keyboard

Success Criteria	WebAIM's Recommendations
2.1.1 Keyboard (Level A)	•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 (Level A)	•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.
2.1.4 Character Key Shortcuts (WCAG 2.1 Level A)	•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 key (Ctrl, Alt, etc.), or only activate the shortcut when an associated interface component or button is focused.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 2.2

Enough Time: Provide users enough time to read and use content

Success Criteria	WebAIM's Recommendations
2.2.1 Timing Adjustable (Level A)	•If a page or application has a time limit, the user is given options to turn off, adjust, or extend that time limit. This is not a requirement for real-time events (e.g., an auction), where the time limit is absolutely required, or if the time limit is longer than 20 hours.
2.2.2 Pause, Stop, Hide (Level A)	•Automatically moving, blinking, or scrolling content (such as carousels, marquees, or animations) that lasts longer than 5 seconds can be paused, stopped, or hidden by the user. •Automatically updating content (e.g., a dynamically-updating news ticker, chat messages, etc.) can be paused, stopped, or hidden by the user or the user can manually control the timing of the updates.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 2.3

Seizures and Physical Reactions: Do not design content in a way that is known to cause seizures or physical reactions.

Success Criteria	WebAIM's Recommendations
2.3.1 Three Flashes or Below Threshold (Level A)	•No page content flashes more than 3 times per second unless that flashing content is sufficiently small and the flashes are of low contrast and do not contain too much red. (See general flash and red flash thresholds)

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 2.4

Navigable: Provide ways to help users navigate, find content, and determine where they are

Success Criteria	WebAIM's Recommendations
2.4.1 Bypass Blocks (Level A)	•A link is provided to skip navigation and other page elements that are repeated across web pages. •A proper heading structure and/or identification of page regions/landmarks may be considered a sufficient technique. Because navigating by headings or regions is not supported in most browsers, WebAIM recommends a "skip" link (in addition to headings and regions) to best support sighted keyboard users.
2.4.2 Page Titled (Level A)	•The web page has a descriptive and informative page title.
2.4.3 Focus Order (Level A)	•The navigation order of links, form elements, etc. is logical and intuitive.
2.4.4 Link Purpose (In Context) (Level A)	•The purpose of each link (or form image button or image map hotspot) can be determined from the link text alone, or from the link text and its context (e.g., surrounding text, list item, previous heading, or table headers). •Links (or form image buttons) with the same text that go to different locations are readily distinguishable.
2.4.5 Multiple Ways (Level AA)	•Multiple ways are available to find other web pages on the site - at least two of: a list of related pages, table of contents, site map, site search, or list of all available web pages.
2.4.6 Headings and Labels (Level AA)	•Page headings and labels for form and interactive controls are informative. Avoid duplicating heading (e.g., "More Details") or label text (e.g., "First Name") unless the structure provides adequate differentiation between them.
2.4.7 Focus Visible (Level AA)	•It is visually apparent which page element has the current keyboard focus (i.e., as you tab through the page, you can see where you are).

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 2.5

Input Modalities: Make it easier for users to operate functionality through various inputs beyond keyboard.

Success Criteria	WebAIM's Recommendations
2.5.1 Pointer Gestures (WCAG 2.1 Level A)	•If multipoint or path-based gestures (such as pinching, swiping, or dragging across the screen) are not essential to the functionality, then the functionality can also be performed with a single point activation (such as activating a button).
2.5.2 Pointer Cancellation (WCAG 2.1 Level A)	•To help avoid inadvertent activation of controls, avoid non-essential down-event (e.g., onmousedown) activation when clicking, tapping, or long pressing the screen. Use onclick, onmouseup, or similar instead. If onmouseup (or similar) is used, you must provide a mechanism to abort or undo the action performed.
2.5.3 Label in Name (WCAG 2.1 Level A)	•If an interface component (link, button, etc.) presents text (or images of text), the accessible name (label, alternative text, aria-label, etc.) for that component must include the visible text.
2.5.4 Motion Actuation (WCAG 2.1 Level A)	•Functionality that is triggered by moving the device (such as shaking or panning a mobile device) or by user movement (such as waving to a camera) can be disabled and equivalent functionality is provided via standard controls like buttons.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 3.1

Readable: Make text content readable and understandable

Success Criteria	WebAIM's Recommendations
3.1.1 Language of Page (Level A)	•The language of the page is identified using the HTML lang attribute (e.g., <html lang="en">).
3.1.2 Language of Parts (Level AA)	•The language of page content that is in a different language is identified using the lang attribute (e.g., <blockquote lang="es">).

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 3.2

Predictable: Make Web pages appear and operate in predictable ways

Success Criteria	WebAIM's Recommendations
3.2.1 On Focus (Level A)	•When a page element receives focus, it does not result in a substantial change to the page, the spawning of a pop-up window, an additional change of keyboard focus, or any other change that could confuse or disorient the user.
3.2.2 On Input (Level A)	•When a user inputs information or interacts with a control, it does not result in a substantial change to the page, the spawning of a pop-up window, an additional change of keyboard focus, or any other change that could confuse or disorient the user unless the user is informed of the change ahead of time.
3.2.3 Consistent Navigation (Level AA)	•Navigation links that are repeated on web pages do not change order when navigating through the site.
3.2.4 Consistent Identification (Level AA)	•Elements that have the same functionality across multiple web pages are consistently identified. For example, a search box at the top of the site should always be labeled the same way.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 3.3

Input Assistance: Help users avoid and correct mistakes

Success Criteria	WebAIM's Recommendations
3.3.1 Error Identification (Level A)	•Required form elements or form elements that require a specific format, value, or length provide this information within the element's label. •Form validation errors are efficient, intuitive, and accessible. The error is clearly identified, quick access to the problematic element is provided, and the user can easily fix the error and resubmit the form.
3.3.2 Labels or Instructions (Level A)	•Sufficient labels, cues, and instructions for required interactive elements are provided via instructions, examples, properly positioned form labels, and/or fieldsets/legends.
3.3.3 Error Suggestion (Level AA)	•If an input error is detected (via client-side or server-side validation), suggestions are provided for fixing the input in a timely and accessible manner.
3.3.4 Error Prevention (Legal, Financial, Data) (Level AA)	•If the user can change or delete legal, financial, or test data, the changes/deletions can be reversed, verified, or confirmed.

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

ACCESSIBILITY OVERVIEW:

Guideline 4.1

Compatible: Maximize compatibility with current and future user agents, including assistive technologies

Success Criteria	WebAIM's Recommendations
4.1.1 Parsing (Level A)	•Significant HTML validation/parsing errors are avoided. Check at http://validator.w3.org/
4.1.2 Name, Role, Value (Level A)	•Markup is used in a way that facilitates accessibility. This includes following the HTML specifications and using forms, form labels, frame titles, etc. appropriately. •ARIA is used appropriately to enhance accessibility when HTML is not sufficient.
4.1.3 Status Messages (WCAG 2.1 Level AA)	•If an important status message is presented and focus is not set to that message, the message must be announced to screen reader users, typically via an ARIA alert or live region

WebAIM: WebAIM's WCAG 2 Checklist (2021). Available at: https://webaim.org/standards/wcag/checklist (Accessed: 30 March 2021).

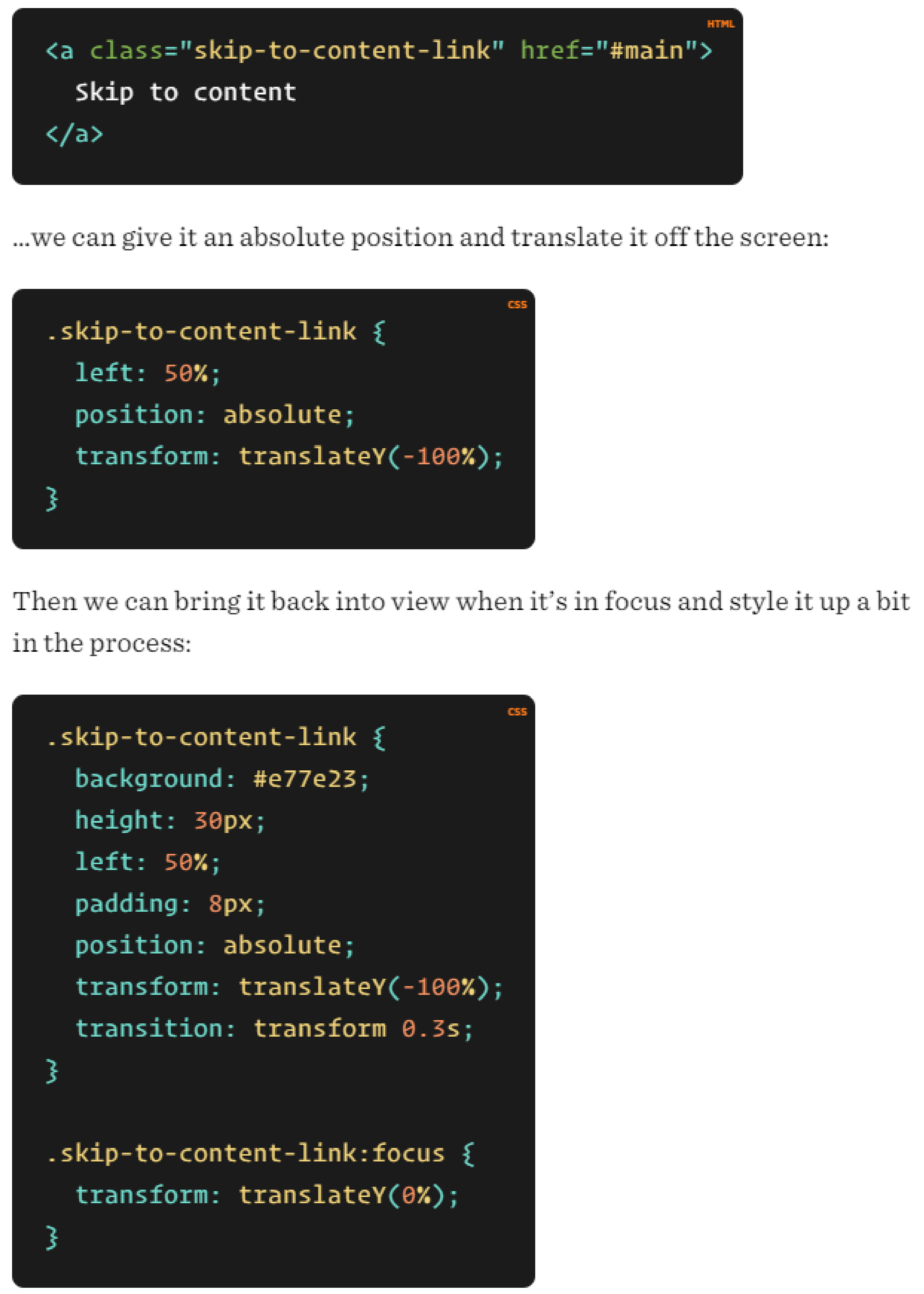
5

ACCESSIBILITY PRINCIPLES AND TECHNIQUES:

Implementation Details

ACCESSIBILITY OVERVIEW:

Skip Content Link



```html
<a class="skip-to-content-link" href="#main">
  Skip to content
</a>
```

…we can give it an absolute position and translate it off the screen:

```css
.skip-to-content-link {
  left: 50%;
  position: absolute;
  transform: translateY(-100%);
}
```

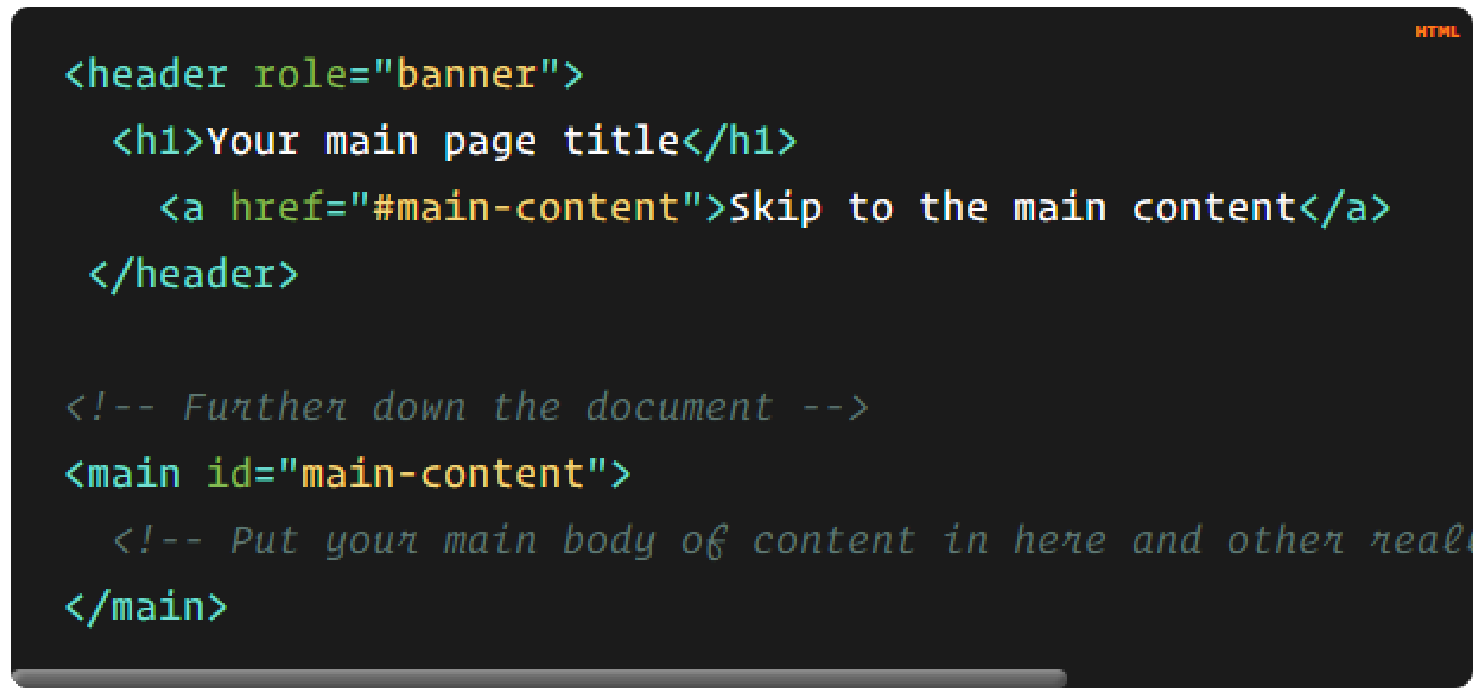
Then we can bring it back into view when it's in focus and style it up a bit in the process:

```css
.skip-to-content-link {
  background: #e77e23;
  height: 30px;
  left: 50%;
  padding: 8px;
  position: absolute;
  transform: translateY(-100%);
  transition: transform 0.3s;
}

.skip-to-content-link:focus {
  transform: translateY(0%);
}
```

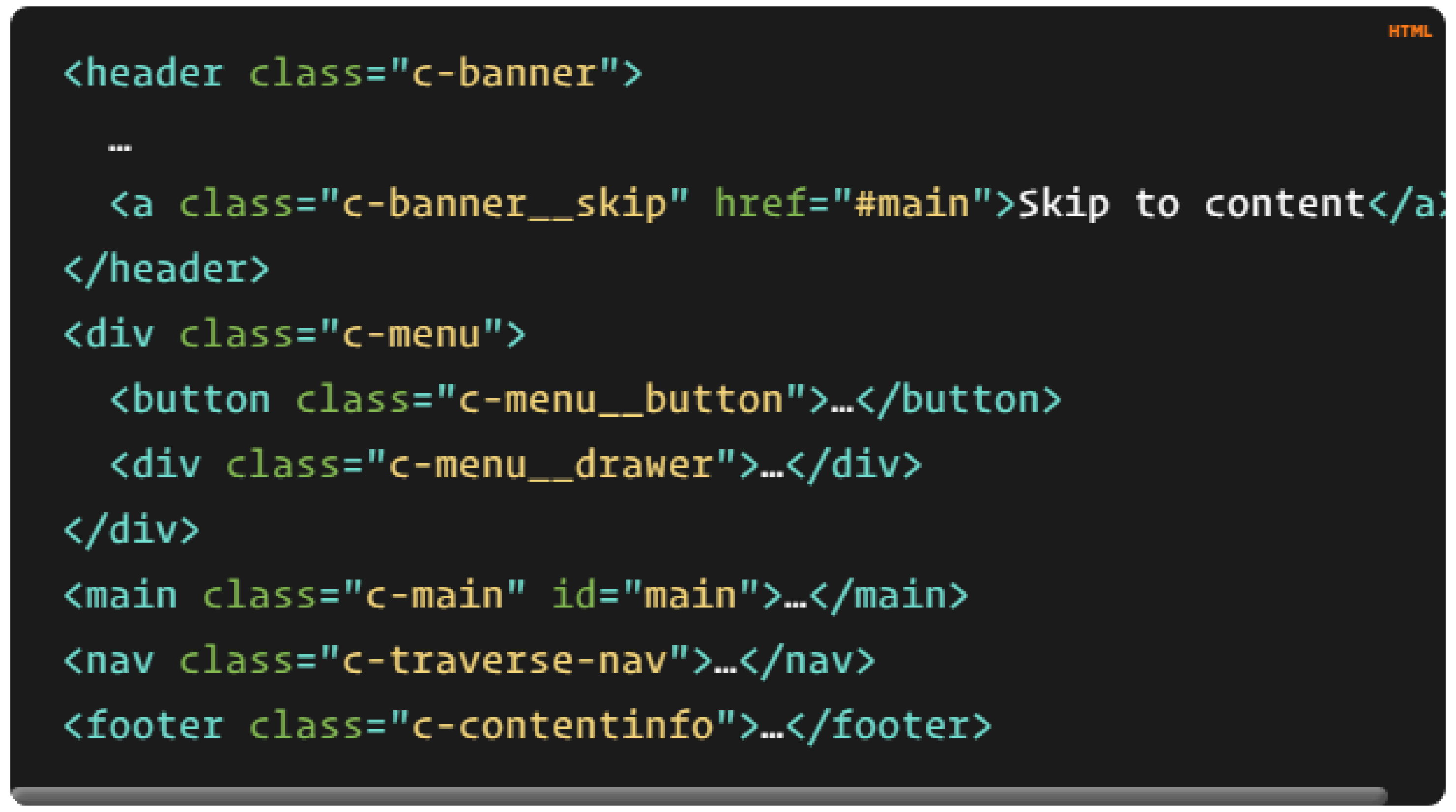
How to Create a "Skip to Content" Link | CSS-Tricks (2020). Available at: https://css-tricks.com/how-to-create-a-skip-to-content-link/ (Accessed: 29 March 2021).

ACCESSIBILITY OVERVIEW:

Document Structure and Semantics


HTML

```html
<header role="banner">
  <h1>Your main page title</h1>
    <a href="#main-content">Skip to the main content</a>
 </header>

<!-- Further down the document -->
<main id="main-content">
  <!-- Put your main body of content in here and other real
</main>
```

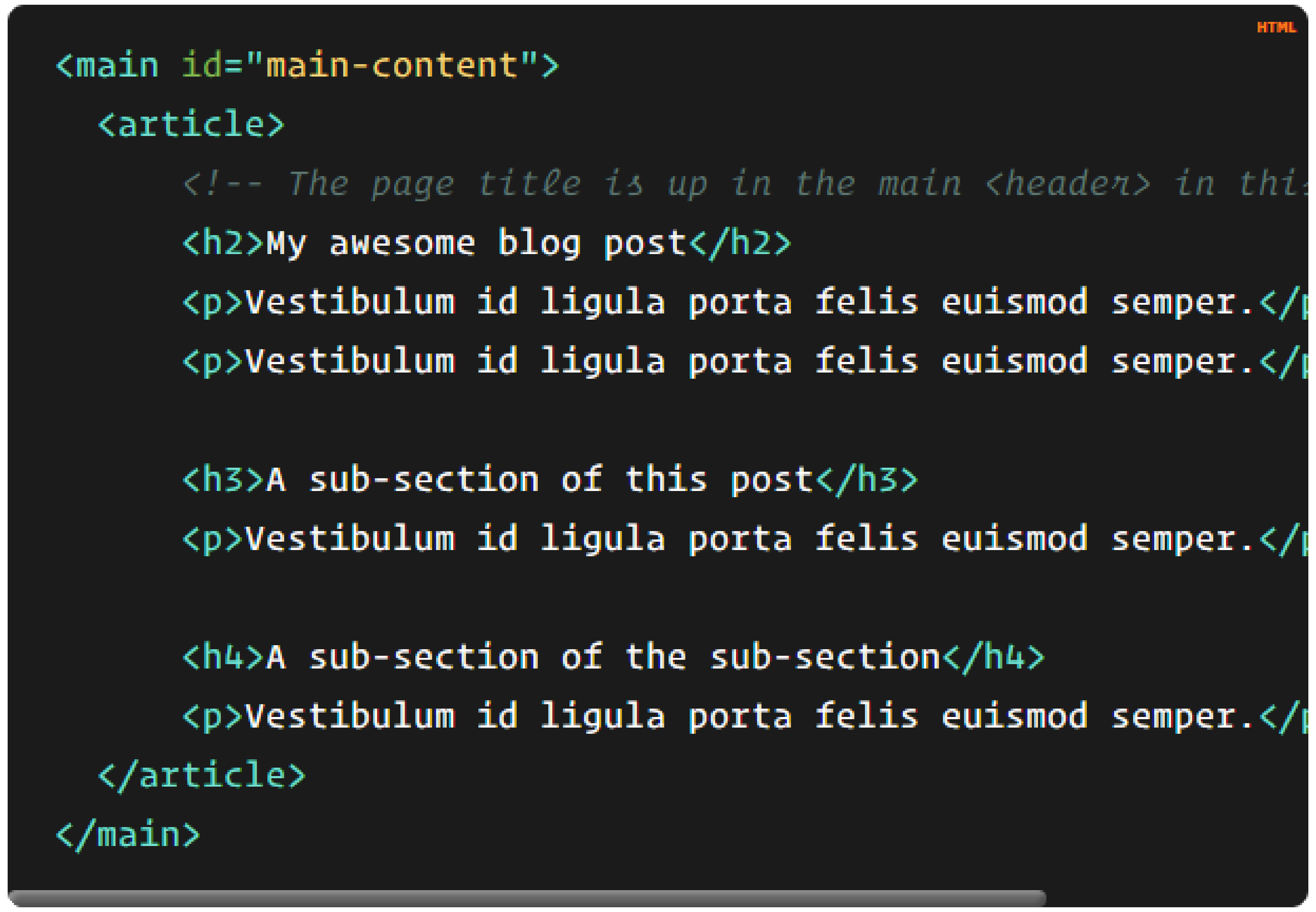

HTML

```html
<header class="c-banner">
  …
  <a class="c-banner__skip" href="#main">Skip to content</a
</header>
<div class="c-menu">
  <button class="c-menu__button">…</button>
  <div class="c-menu__drawer">…</div>
</div>
<main class="c-main" id="main">…</main>
<nav class="c-traverse-nav">…</nav>
<footer class="c-contentinfo">…</footer>
```

Small Tweaks That Can Make a Huge Impact on Your Website's Accessibility | CSS-Tricks (2018). Available at: https://css-tricks.com/small-tweaks-can-make-huge-impact-websites-accessibility/ (Accessed: 29 March 2021).

ACCESSIBILITY OVERVIEW:

Heading Hierarchy



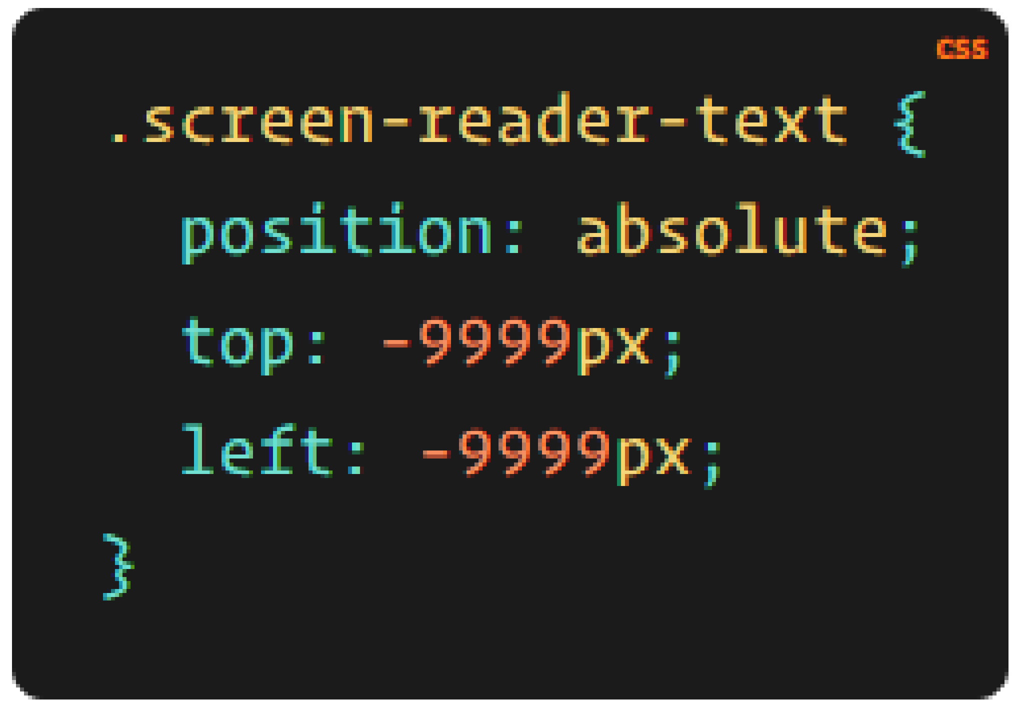
HTML

```html
<main id="main-content">
  <article>
      <!-- The page title is up in the main <header> in thi
      <h2>My awesome blog post</h2>
      <p>Vestibulum id ligula porta felis euismod semper.</
      <p>Vestibulum id ligula porta felis euismod semper.</

      <h3>A sub-section of this post</h3>
      <p>Vestibulum id ligula porta felis euismod semper.</

      <h4>A sub-section of the sub-section</h4>
      <p>Vestibulum id ligula porta felis euismod semper.</
  </article>
</main>
```

Improving the Accessibility of 24 ways | CSS-Tricks (2018). Available at: https://css-tricks.com/improving-accessibility-24-ways/ (Accessed: 29 March 2021).

ACCESSIBILITY OVERVIEW:

Accessibility/SEO Friendly CSS Hiding

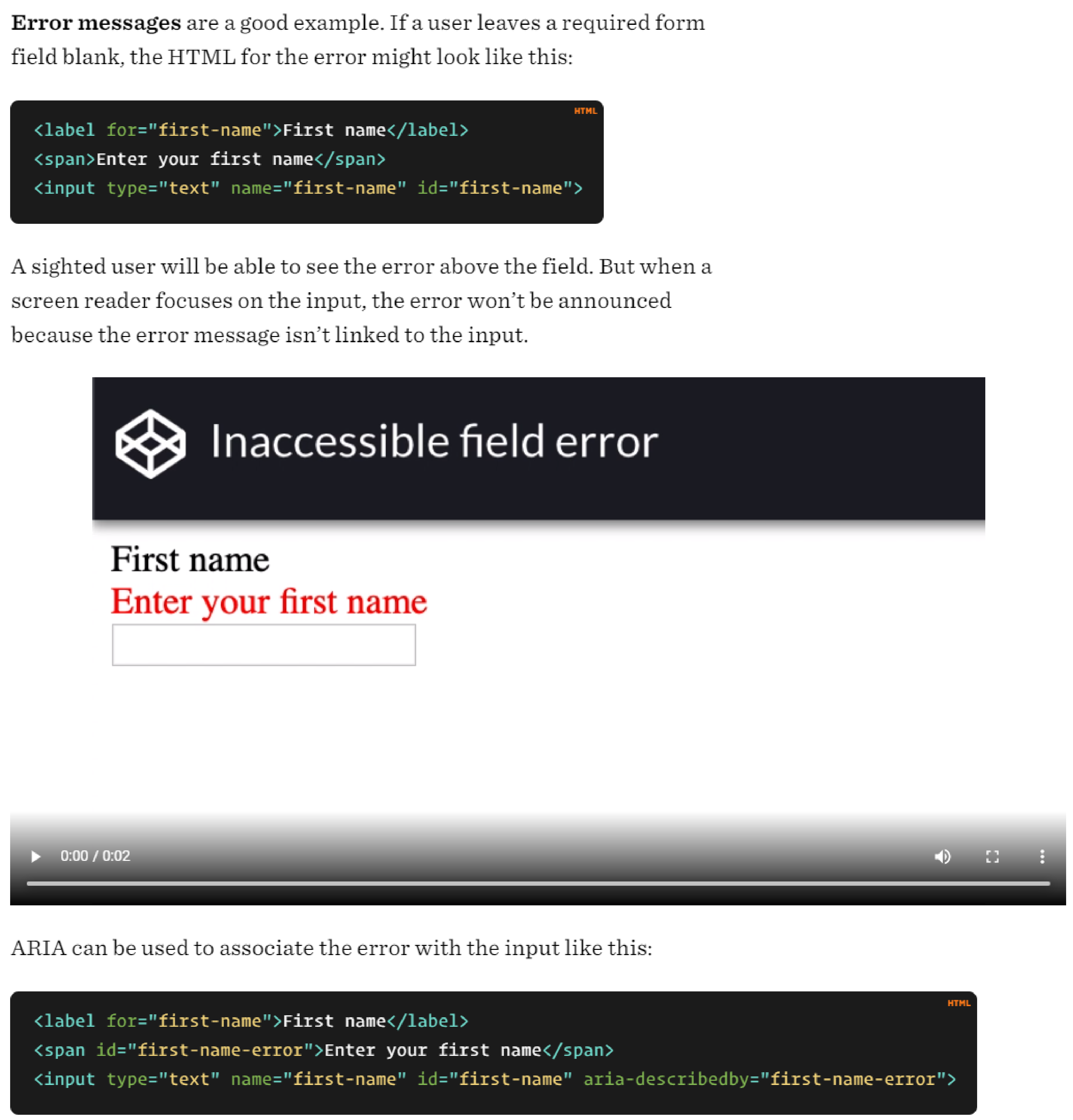


```css
.screen-reader-text {
  position: absolute;
  top: -9999px;
  left: -9999px;
}
```

Inclusively Hidden | *CSS-Tricks* (2019). Available at: https://css-tricks.com/inclusively-hidden/ (Accessed: 29 March 2021).

ACCESSIBILITY OVERVIEW:

Semantic HTML and ARIA



Error messages are a good example. If a user leaves a required form field blank, the HTML for the error might look like this:

```html
<label for="first-name">First name</label>
<span>Enter your first name</span>
<input type="text" name="first-name" id="first-name">
```

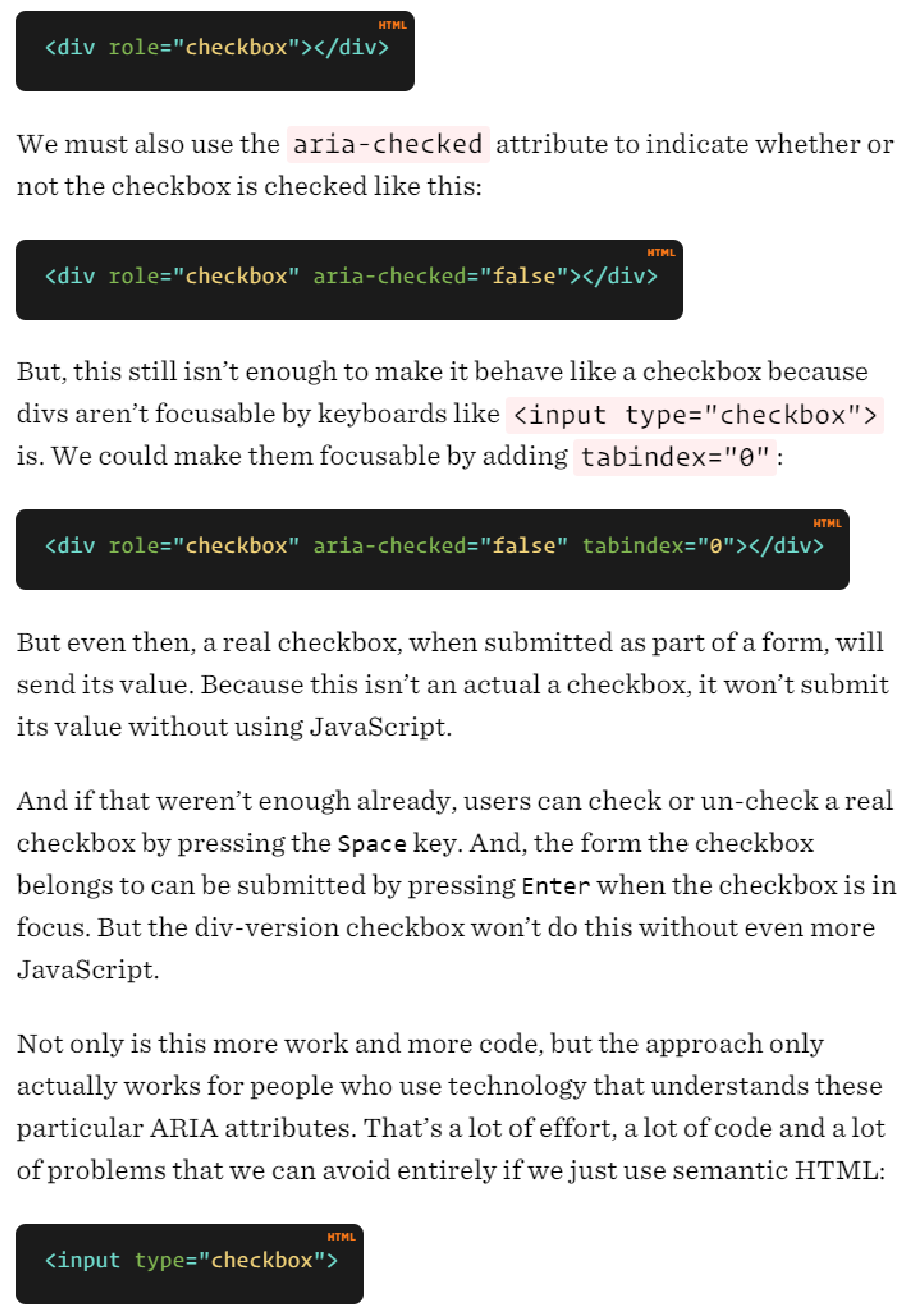
A sighted user will be able to see the error above the field. But when a screen reader focuses on the input, the error won't be announced because the error message isn't linked to the input.

ARIA can be used to associate the error with the input like this:

```html
<label for="first-name">First name</label>
<span id="first-name-error">Enter your first name</span>
<input type="text" name="first-name" id="first-name" aria-describedby="first-name-error">
```

Why, How, and When to Use Semantic HTML and ARIA | CSS-Tricks (2019). Available at: https://css-tricks.com/why-how-and-when-to-use-semantic-html-and-aria/ (Accessed: 29 March 2021).

Avoid using ARIA to fix unsemantic HTML



```html
<div role="checkbox"></div>
```

We must also use the `aria-checked` attribute to indicate whether or not the checkbox is checked like this:

```html
<div role="checkbox" aria-checked="false"></div>
```

But, this still isn't enough to make it behave like a checkbox because divs aren't focusable by keyboards like `<input type="checkbox">` is. We could make them focusable by adding `tabindex="0"`:

```html
<div role="checkbox" aria-checked="false" tabindex="0"></div>
```

But even then, a real checkbox, when submitted as part of a form, will send its value. Because this isn't an actual a checkbox, it won't submit its value without using JavaScript.

And if that weren't enough already, users can check or un-check a real checkbox by pressing the `Space` key. And, the form the checkbox belongs to can be submitted by pressing `Enter` when the checkbox is in focus. But the div-version checkbox won't do this without even more JavaScript.

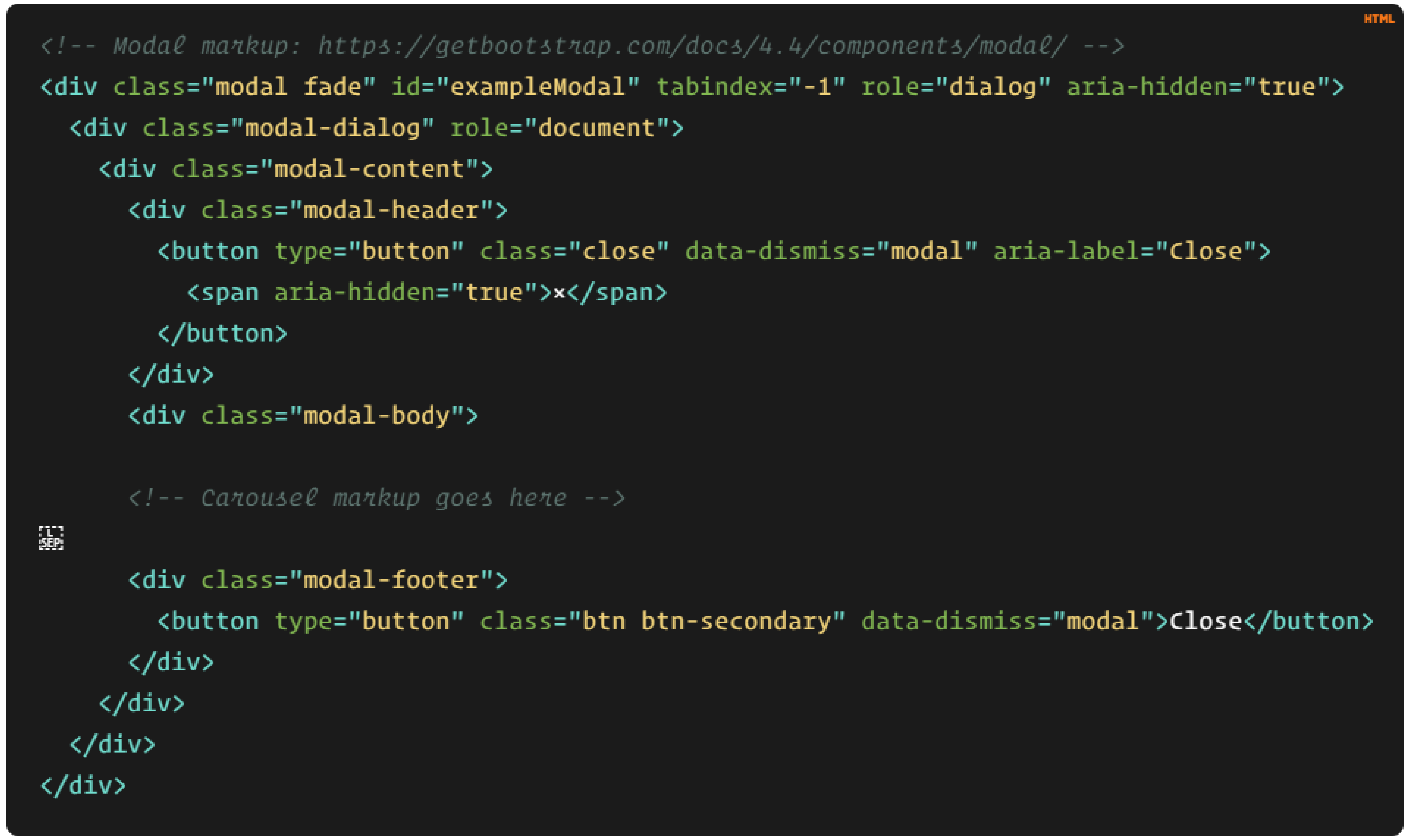
Not only is this more work and more code, but the approach only actually works for people who use technology that understands these particular ARIA attributes. That's a lot of effort, a lot of code and a lot of problems that we can avoid entirely if we just use semantic HTML:

```html
<input type="checkbox">
```

How to Create a "Skip to Content" Link | CSS-Tricks (2020). Available at: https://css-tricks.com/how-to-create-a-skip-to-content-link/ (Accessed: 29 March 2021).

ACCESSIBILITY OVERVIEW:

Open-Source Accessibility



HTML

```html
<!-- Modal markup: https://getbootstrap.com/docs/4.4/components/modal/ -->
<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">×</span>
        </button>
      </div>
      <div class="modal-body">

      <!-- Carousel markup goes here -->

      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
      </div>
    </div>
  </div>
</div>
```

Libraries

- React Material-UI
- Twitter Bootstrap

Tools

- dequelabs/axe-core
- GoogleChrome/lighthouse
- microsoft/accessibility-insights-web

Reference Sites

- Twitter
- Google Accessibility Site

Mark Otto, a. (2021) Introduction, Getbootstrap.com. Available at: https://getbootstrap.com/docs/5.0/getting-started/introduction/ (Accessed: 29 March 2021).

ACCESSIBILITY OVERVIEW:

Continued Learning

This presentation only covered Website Accessibility using WCAG 2.1 Level A/AA

- WCAG 2.1 Level AAA
- Mobile Native Accessibility (iOS, Android, etc.)
- Taking Accessibility Beyond Compliance