
Web Best Practices and Accessibility

—

Lecture 2

—

Learning Objectives

After completing this lesson, you should be able to explain about

- web best practices

- web accessibility

Web best practices

Cross-browser compatibility

the practice of trying to make sure your webpage works across as many devices as possible

Web best practices

Performance

getting websites to load as quickly as possible, but also making them intuitive and easy to use so that users don't get frustrated and go somewhere else

Web best practices

Responsive web design

the practice of making your functionality and layouts flexible so they can automatically adapt to different browsers and device sizes

Web best practices

Accessibility

making your websites usable by as many different kinds of people as possible

includes people with visual impairments, hearing impairments, cognitive disabilities, or physical disabilities

also includes young or old people, people from different cultures, people using mobile devices, or people with unreliable or slow network connections

Web best practices

Internationalization

making websites usable by people from different cultures, who speak different languages to your own

There are technical considerations here (such as altering your layout so that it still works OK for right-to-left, or even vertical languages)

Web best practices

Privacy & Security

Privacy refers to allowing people to go about their business privately and not spying on them or collecting more of their data than you absolutely need to

Security refers to constructing your website in a secure way so that malicious users cannot steal information contained on it from you or your use

What is Accessibility

Accessibility is the practice of **making your websites usable by as many people as possible** including

- people with disabilities
- people using mobile devices
- people with slow network connections

Providing accessible sites is part of the law in some countries

Example: Which one is more accessible

jo@example.com

Pay with

Select payment method ▼

Ship to

Choose address ▼

Remember details?

☒

Subtotal

\$10.00

Shipping

\$0.00

Tax

\$0.90

Total

\$10.90

Place Order

jo@example.com

Subtotal

\$10.00

Shipping

\$0.00

Tax

\$0.90

Total

\$10.90

Pay with:

Select payment method ▼

Ship to:

Choose address ▼

☒ Remember details?

Place Order

Disabilities: Visual Impairment

People with blindness, low-level vision, and color blindness

People with visual impairments use

screen magnifiers that are either physical magnifiers or software zoom capabilities

screen readers, which is software that reads digital text aloud

Softwares for visual impaired people

NVDA, ChromeVox, Orca, VoiceOver, Narrator, TalkBack

Disabilities: Hearing Impairment

To provide access, textual **alternatives** must be provided

Videos should be manually **captioned**

Transcripts should be provided for audio content

Disabilities: Mobility Impairment

Some people might have difficulty making the exact hand movements required to use a mouse

To provide access controls should be accessible by the keyboard

Disabilities: Cognitive Impairment

To provide access for people with cognitive impairments consider

- Delivering content in more than one way, such as by text-to-speech or by video
- Easily understood content, such as text written using plain-language standards
- Focusing attention on important content

Disabilities: Cognitive Impairment

To provide access for people with cognitive impairments consider

- Minimizing distractions, such as unnecessary content or advertisements
- Consistent web page layout and navigation
- Familiar elements, such as underlined links blue when not visited and purple when visited

Disabilities: Cognitive Impairment

To provide access for people with cognitive impairments consider

- Dividing processes into logical, essential steps with progress indicators
- Website authentication as easy as possible without compromising security
- Making forms easy to complete, such as with clear error messages and simple error recovery

Web Content Accessibility Guidelines (WCAG)

WCAG is organized around four principles often called by the acronym **POUR**:

Perceivable: Can users perceive the content?

Just because something is perceivable with one sense, such as sight, that doesn't mean that all users can perceive it

Web Content Accessibility Guidelines (WCAG)

WCAG is organized around four principles often called by the acronym **POUR**:

Operable: Can users use UI components and navigate the content?

For example, something that requires a hover interaction cannot be operated by someone who can't use a mouse or touch screen

Web Content Accessibility Guidelines (WCAG)

WCAG is organized around four principles often called by the acronym **POUR**:

Understandable: Can users understand the content?

Can users understand the interface and is it consistent enough to avoid confusion?

Web Content Accessibility Guidelines (WCAG)

WCAG is organized around four principles often called by the acronym **POUR**:

Robust: Can the content be consumed by a wide variety of user agents (browsers)?

Does it work with assistive technology?

Assistive technology

Assistive technology is an umbrella term for devices, software, and tools that help any person with a disability complete a task

Demo

<http://udacity.github.io/ud891/lesson3-semantics-built-in/02-chromevox-lite/>

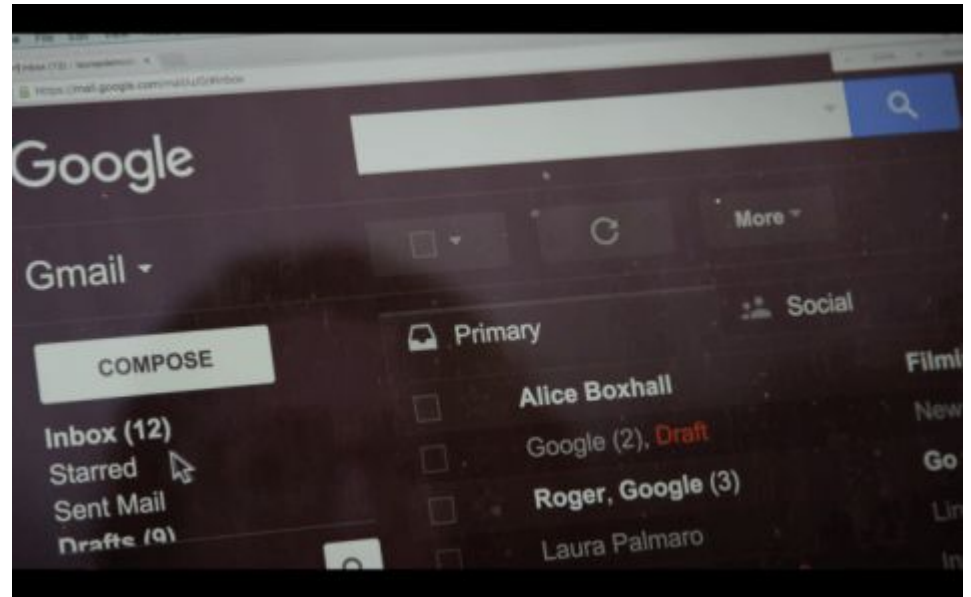
Example Assistive Technologies

Braille Reader



Example Assistive Technologies

High Contrast Mode



Example Assistive Technologies

Eye tracking device



Example Assistive Technologies

Screen Caption



Assess the accessibility of a web page

<https://www.w3.org/WAI/test-evaluate/preliminary/>

Designing for Web Accessibility

Make your user interface design and visual design more accessible to people with disabilities

<https://www.w3.org/WAI/tips/designing/>

Designing for Web Accessibility

Provide sufficient contrast between foreground and background

Example: Contrast ratio

✖ Insufficient

Some people cannot read text if there is not sufficient contrast between the text and background. For others, bright colors (high luminance) are not readable; they need low luminance.

✔ Sufficient

Some people cannot read text if there is not sufficient contrast between the text and background. For others, bright colors (high luminance) are not readable; they need low luminance.

Designing for Web Accessibility

Don't use color alone to convey information

Example: Using color to convey meaning

✖ Color only

Required fields are in red

Name

Email

✔ Color and symbol

Required fields are in red and marked with an *

Name

Email *

Designing for Web Accessibility

Don't use color alone to convey information

Example: Refer to something using color alone

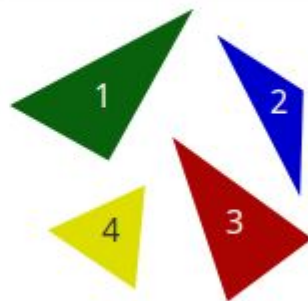
✖ Color only



Which is the right-angled triangle?

- ☐ Green
- ☐ Blue
- ☐ Red
- ☐ Yellow
- ☐ Don't know

✔ Color and number



Which is the right-angled triangle?

- ☐ Green (1)
- ☐ Blue (2)
- ☐ Red (3)
- ☐ Yellow (4)
- ☐ Don't know

Designing for Web Accessibility

Ensure that interactive elements are easy to identify

Example: Unique styles for different link states

✔ Style links to stand out from text

Some people can't use a mouse and use only a [keyboard to navigate](#) through web pages.

It is important that users can reach all interactive elements using the keyboard, and that it is clear which element has focus.

Visible keyboard focus could be a border or highlight that moves as you tab through the web page.

✔ Mouse hover style

[keyboard to navigate](#)



✔ Keyboard focus style

[keyboard to navigate](#)

✔ Touch or click style

[keyboard to navigate](#)



Designing for Web Accessibility

Provide clear and consistent navigation options

- Ensure that navigation across pages within a website has consistent naming, styling, and positioning
- Provide more than one method of website navigation, such as a site search or a site map
- Help users understand where they are in a website or page by providing orientation cues, such as breadcrumbs and clear headings

Designing for Web Accessibility

Ensure that form elements include clearly associated labels

✓ Example: Labels and input fields associated by proximity

Add a comment

Your E-mail

☐ I am happy for you to contact me

Your Website

Comment

Designing for Web Accessibility

Provide easily identifiable feedback



✔ Example: Using error list, icon, and background color to make errors stand out

Please correct the following errors:

1.  [Email address is invalid](#)
2.  [A Comment is required](#)

Add a comment

Required fields are in red and marked with an *

Name	<input type="text" value="Superbear"/>
 E-mail *	<input type="text" value="superbear@@hq.example.com"/>
Website	<input type="text"/>
 Comment *	<div><div></div></div>

Designing for Web Accessibility

Use headings and spacing to group related content

Example: Spacing highlights relationship between content

✖ Little spacing and unclear relationship

Main heading

Horizontal lines representing text content.

Sub heading

Horizontal lines representing text content.

Sub heading

Horizontal lines representing text content.



✔ More spacing and clearer relationship

Main heading

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.



Sub heading

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

Sub heading

Horizontal lines representing text content.

Horizontal lines representing text content.

Horizontal lines representing text content.

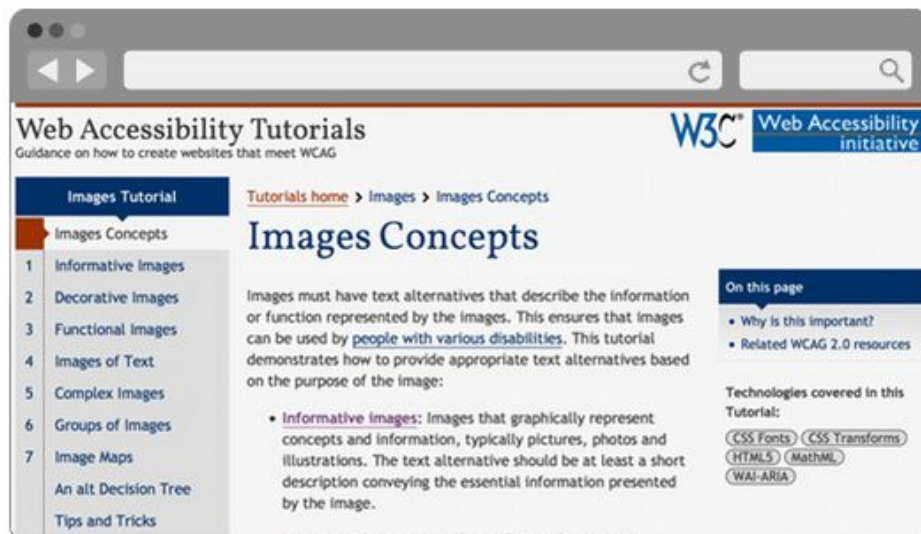
Horizontal lines representing text content.

Horizontal lines representing text content.

Designing for Web Accessibility

Create designs for different viewport sizes

Example: Content and navigation adapt to smaller mobile screen



Designing for Web Accessibility

Include image and media alternatives in your design

Example: Design includes links to a transcript and to an audio described video

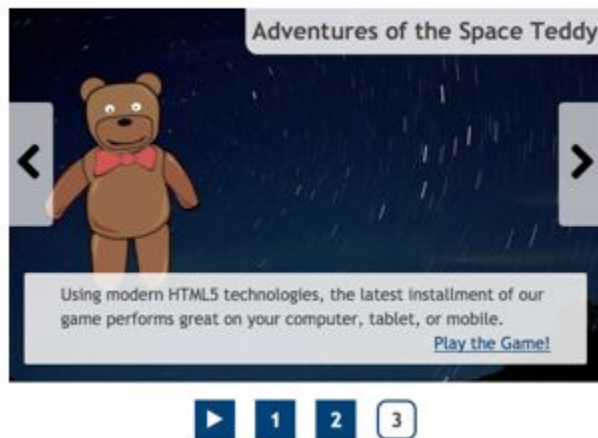


[Transcript](#) | 

Designing for Web Accessibility

Provide controls for content that starts automatically

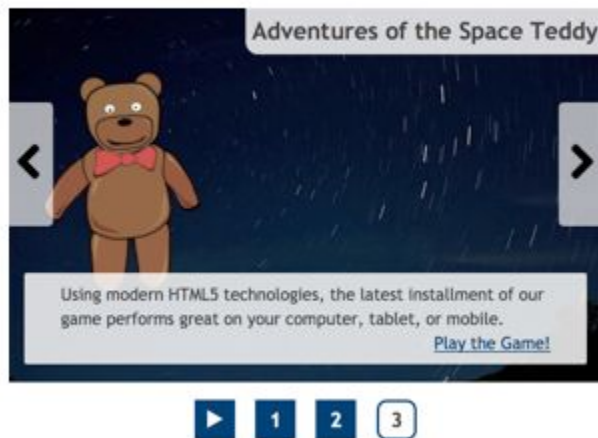
Example: Show play/stop and selection controls in carousel design



Designing for Web Accessibility

Provide controls for content that starts automatically

Example: Show play/stop and selection controls in carousel design



Developing for Web Accessibility

Associate a `label` with every form control

Example: Using `for` and `id` attributes

Rendered

Username

Code Snippet

```
<label for="username">Username</label>
<input id="username" type="text"
name="username">
```


Developing for Web Accessibility

Include alternative text for images

- Ensure that alternative text for images is added to all informational and functional images
- Use empty alternative text, `alt=""` for decorative images, or include them in the CSS instead

Developing for Web Accessibility

Identify page language and language changes

- Indicate the primary language of every page by using the **lang** attribute in the **html** tag, for example `<html lang="en">`
- Use the **lang** attribute on **specific elements** when the language of the element differs from the rest of the page.

Developing for Web Accessibility

Use markup to convey meaning and structure

- Use appropriate markup for headings, lists, tables, etc
- **HTML5** provides additional elements, such as `<nav>` and `<aside>`, to better structure your content
- **WAI-ARIA** roles can provide additional meaning, for example, using `role="search"` to identify search functionality.

Developing for Web Accessibility

Use markup to convey meaning and structure

- Use appropriate markup for headings, lists, tables, etc
- **HTML5** provides additional elements, such as `<nav>` and `<aside>`, to better structure your content
- **WAI-ARIA** roles can provide additional meaning, for example, using `role="search"` to identify search functionality.

Developing for Web Accessibility

Help users avoid and correct mistakes

Provide clear instructions, error messages, and notifications to help users complete forms on your site. When an error occurs:

- Help users find where the problem is
- Provide specific, understandable explanations
- Suggest corrections

Developing for Web Accessibility

Help users avoid and correct mistakes

Example: Australian phone number field with forgiving validation

Rendered

Phone

For example, (02) 1234 1234

Code Snippet

```
<label for="phone">Phone</label>
<input id="phone" name="phone" type="tel"
      pattern="^\(?(?0[1-9]{1}\)?)?[0-9
-]*$"
      aria-describedby="phone-desc">
<p id="phone-desc">For example, (02) 1234
1234</p>
```

Developing for Web Accessibility

Help users avoid and correct mistakes

Example: Australian phone number field with forgiving validation

Rendered

Phone

For example, (02) 1234 1234

Code Snippet

```
<label for="phone">Phone</label>
<input id="phone" name="phone" type="tel"
      pattern="^\(?(?0[1-9]{1}\)?)?[0-9
-]*$"
      aria-describedby="phone-desc">
<p id="phone-desc">For example, (02) 1234
1234</p>
```

Developing for Web Accessibility

Reflect the reading order in the code order

✔ Heading marks the start of the section

```
<h3>Space trainers</h3>


<p>Space...</p>
<a href="...">Add to cart</a>
```

Example: Reflecting the logical reading order in the code



Space trainers

Space trainer for a classic and stylish look.

 Add to cart

✖ Image before heading may be missed

```

<h3>Space trainers</h3>
<p>Space...</p>
<a href="...">Add to cart</a>
```


Developing for Web Accessibility

Write code that adapts to the user's technology

Use **responsive design** to adapt the display to different zoom states and viewport sizes, such as on mobile devices and tablets

When font size is increased by at least 200%, avoid horizontal scrolling and prevent any clipping of content

Developing for Web Accessibility

Provide meaning for non-standard interactive elements

Use **WAI-ARIA** to provide information on function and state for custom widgets, such as accordions and custom-made buttons

For example, `role="navigation"` and `aria-expanded="true"`

Developing for Web Accessibility

Ensure that all interactive elements are keyboard accessible

Think about keyboard access, especially when developing interactive elements, such as menus, mouseover information, collapsible accordions, or media players

Use `tabindex="0"` to add an element that does not normally receive focus, such as `<div>` or ``, into the navigation order when it is being used for interaction

Use scripting to capture and respond to keyboard events

Developing for Web Accessibility

Avoid CAPTCHA where possible

CAPTCHAs create problems for many people

There are other means of verifying that user input was generated by a human that are easier to use, such as automatic detection or interface interactions

WAI-ARIA

WAI-ARIA, the **W**eb **A**ccessibility **I**nitiative - **A**ccessible **R**ich **I**nternet **A**pplications suite

defines a way to make Web content and Web applications more accessible to people with disabilities

<https://www.w3.org/WAI/standards-guidelines/aria/>

WAI-ARIA

WAI-ARIA provides a framework for **adding attributes** to
identify features for user **interaction**,
how they **relate** to each other, and
their current **state**

WAI-ARIA

WAI-ARIA describes navigation techniques to mark regions and common web structures as

menus, primary content, secondary content, banner information, and other types of Web structures

WAI-ARIA

WAI-ARIA includes technologies to map controls, live regions, and events to accessibility application programming interfaces (APIs), including custom controls used for rich Internet applications