AccessComputing

# 30 Web Accessibility Tips

by Terrill Thompson, Technology Accessibility Specialist

This publication provides thirty web accessibility tips that can be used by web designers, developers, or content authors to guide them in creating or deploying web-based resources that are fully accessible to all users. This list is not intended to replace or map to formal standards such as the World Wide Web Consortium's (W3C's) Web Content Accessibility Guidelines (*https://www.w3.org/WAI/standards-guidelines/wcag/*).

Each of these tips is described in more detail at ***uw.edu/accesscomputing/tips***. Have suggestions for how this list can be improved? Please send your ideas to *accesscomp@uw.edu*.

1. **Add proper alt text to images.**
   Use alt text to provide access to the content of images for individuals who cannot see them, including people using screen readers or Braille output devices. Alt text is supported by most document formats, including HTML, Microsoft Word, and Adobe PDF.

2. **Use headings properly.**
   Use headings and subheadings to form an outline of the page. Do not skip heading levels. They help non-visual users (including search engines) to understand how the page is organized, and make it easy for screen reader users to navigate.

3. **Create accessible PDFs.**
   Use "tagged PDF", the only type of PDF that supports headings and alt text for images. Use the PDF Accessibility Checker that's provided in Adobe Acrobat.

4. **Know when to use PDF.**
   PDF preserves a document's appearance across operating systems and devices. If this characteristic is not essential, consider using an alternate format such as HTML.

5. **Use ARIA landmarks.**
   The Accessible Rich Internet Applications (ARIA) W3C specification makes it possible to produce accessible interactive web applications. One easy entry into ARIA is landmark roles. Simply add an HTML attribute for one of the eight landmark roles (e.g., role="navigation", role="main") and users will be able to jump directly to a section of the page with a single keystroke. Alternatively, Use HTML semantic elements that map to ARIA roles (e.g., **navigation**, **main**).

6. **Add labels to form fields.**
   Use the HTML **label** element to screen reader users will know which labels or prompts are associated with which form fields.

7. **Group related form fields together.**
   In HTML, wrap groups of checkboxes or radio buttons in a **fieldset** element, and wrap the question or prompt that applied to them all in a **legend** element.

8. **Markup tables appropriately.**
   Use HTML markup properly to communicate relationships between column and row headers and the data cells within their scope.

9. **Identify language of text.**
   Since some screen readers are multi-lingual, use markup to identify the default language of a document and of any text that deviates from the default.

10. **Use a color contrast checker.**
    Be sure foreground and background have adequate contrast. There are many free tools that can help with this.

11. **Avoid using tiny fonts.**
    Since users may be unaware they can increase font size using browser hot keys, use a reasonably large font size by default; then, users can make it smaller if desired. Note that a font size of **1em** uses the default browser font size, therefore is an ideal choice for most text, thereby honoring users' preferences and expectations.

12. **Respect white space.**
    Provide plenty of space between lines and blocks of text. This helps many users to more easily track text horizontally, and generally makes text easier to read.

13. **Provide visible indication of focus.**
    In CSS, use **:hover** to make the page come alive, responding to user's mouse movements. For people who aren't using a mouse, provide the same functionality using **:focus**.

14. **Use text, not pictures of text.**
    Use text instead of pictures of text, and control its appearance using CSS. Pictures of text become blurry when enlarged, take longer to download, and are inefficient for the website author to edit.

15. **Think twice about the words you choose.**
    Keep your content simple to read and understand. Often web authors use larger words and longer sentences than is really necessary.

16. **Caption video.**
    Captions provide benefits to all users. There are many free, easy-to-use tools available that support the processes of transcribing and captioning videos.

17. **Describe video.**
    For people are unable to see video, create a script that includes brief descriptions of important visual content, then provide that as a separate description track, either as timed text or recorded narration.

18. **Provide a transcript.**
    Provide a transcript for video and audio so individuals who are deaf-blind and those with low Internet bandwidth can access the content. Transcripts benefit all users by allowing them to access content quickly.

19. **Choose media players that support accessibility.**
    Ask questions like: Does this player support closed captions? Does it support description? Can it be operated without a mouse? Are buttons and controls accessible to screen reader users?

20. **Choose a website navigation menu that works for all users.**
    Ask questions like: Can this menu be operated by keyboard alone? If so, is doing so efficient or does it require dozens or hundreds of keystrokes?

21. **Choose JavaScript widgets that support accessibility.**
    An accessible interactive widget is one that can be operated with the keyboard alone, uses ARIA to communicate with assistive technology (AT), and degrades gracefully for users who don't have the latest AT.

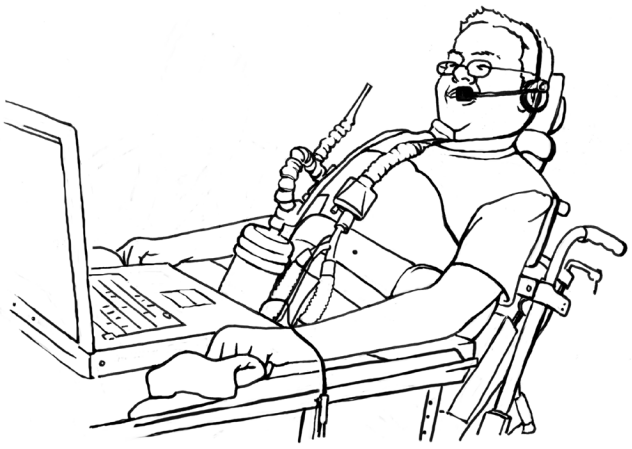
22. **Test Javascript Widgets.**
    Perform due diligence when choosing a widget to add to your web page. Review documentation to see if accessibility is addressed, test widgets using a keyboard alone or using AT, and ask users with disabilities to use them.

23. **Choose Learning Management Systems (LMS) and Content Management Systems (CMS) that support accessibility.**
    Make sure that your organization's LMS or CMS supports accessibility. Make sure there are accessible themes, plug-ins, and modules available that will meet your needs.

24. **Test web pages using a keyboard.**
    Try navigating the web page and controlling all its features using the tab key on a keyboard; don't touch the mouse. This simple test is typically a good indicator of accessibility.

25. **Test pages using high contrast color schemes.**
    All major operating systems have high contrast color schemes available. When it is enabled, make sure that all important page content is still visible.

26. **Test pages with assistive technologies.**
    There are free screen readers and other AT available that can be used for testing. You don't have to become an expert user, but simple tests with AT can provide valuable insights into whether certain features on a web page might present accessibility problems.



27. **Test pages on mobile devices.**
    Growing numbers of users, including users with disabilities, are accessing the web using phones, tablets, and other mobile devices. Test your website using mobile devices, and when doing so, be sure to check for accessibility.

28. **Ask vendors specific questions about the accessibility of their products.**
    "Is your product accessible?" is not a good question. Ask them to demonstrate how their product can be used without a mouse or with a screen reader. Ask about their methods for testing accessibility.

29. **Demand accessibility!**
    If a product is not accessible, avoid buying it, using it, or supporting it. Work to implement policies on your campus that require IT purchases to be accessible. If it is required that an inaccessible product be used because it is the only current alternative, put the vendor on notice that you expect to purchase an accessible alternative when it becomes available.

30. **Get involved!**
    There are many communities of practice that are working together on making the web (and world) more accessible. Together we can make a difference!

## About *AccessComputing*

The Department of Computer Science and Engineering and DO-IT (Disabilities, Opportunities, Internetworking and Technology) at the University of Washington lead the *AccessComputing* Project for the purpose of increasing the participation of people with disabilities in computing careers nationwide. *AccessComputing* partners include Gallaudet University, Landmark College, Rochester Institute of Technology, Microsoft, the NSF Regional Alliances for Persons with Disabilities in STEM. Collaborators represent education, industry, government, and professional organizations nationwide.

For further information, to be placed on the mailing list, request materials in an alternate format, or to make comments or suggestions about DO-IT publications or web pages, contact:

University of Washington
Box 354842
Seattle, WA 98195-4842
*accesscomp@uw.edu*
*www.uw.edu/accesscomputing/*
206-685-DOIT (3648) (voice/TTY)
888-972-DOIT (3648) (toll free voice/TTY)
509-328-9331 (voice/TTY) Spokane
206-221-4171 (fax)

*AccessComputing*
Dr. Richard Ladner, PI
Dr. Sheryl Burgstahler, Co-PI

Grants and gifts fund DO-IT publications, videos, and programs to support the academic and career success of people with disabilities. Contribute today by sending a check to DO-IT, Box 354842, University of Washington, Seattle, WA 98195-4842.

*Your gift is tax deductible as specified in IRS regulations. Pursuant to RCW 19.09, the University of Washington is registered as a charitable organization with the Secretary of State, State of Washington. For more information call the Office of the Secretary of State, 800-322-4483.*

**Acknowledgment**
*AccessComputing* is supported by the National Science Foundation under Grant #CNS-0540615, #CNS-0837508, and #CNS-1042260. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.




**University of Washington**
Computer Science and Engineering
DO-IT
UW Information Technology