ACCESS university

Top 10 Takeaways

Web Accessibility Advanced

1 CSS and Typography

Using CSS's content property with the :before or :after pseudo class for valuable information will not be rendered to those using their own style sheets. Place the content directly into the HTML. Ensure the reading order of content and elements are correct when viewed without style sheets. Reading Order is based on order within the HTML content. Users with Assistive Technologies interact following the HTML order, not the CSS Order. Ensure text can be resized. Users may use the zoom feature in web browsers to enlarge text. If using fixed sizes, content and other elements may, not scale correctly or at all, or get cut off at lower resolutions, causing users to have to scroll horizontally across the page.

2 Background Images

Ensure there are equivalents for background images that convey meaning. Background images that convey meaning may be lost for those with assistive technologies. Providing off screen text will not suffice as the text is not shown to users with these assistive technologies. The alternative for the background image must be accessible to users of assistive technology, and visually accessible to users with low vision who cannot view background images but that may not be using assistive technology. It is best to avoid the use of CSS background images that convey meaning or sensory experience.

3 Advanced Keyboard Accessibility

A keyboard event handler is a function that responds to a user input event, such as a mouse click or key press. Event handlers that rely solely on mouse-related events, such as `onMouseOver`, cannot be triggered by users who do not use a mouse. Device independent event handlers, where supported by the assistive technology, can also be used to meet this best practice. For elements that do not naturally receive keyboard access, use device independent event handlers. Some situations require dual event handlers, such as `onclick` and `onkeyup`.

4 Focus Control

Avoid forced focus changes that are not initiated by the user. Users expect focus to be shifted when the Tab key, Shift+Tab keys, Enter key, arrow keys or the Spacebar is pressed on a user actionable control. When focus is moved without user interaction, users may be unaware the focus changed, or the user may need to press many keystrokes to return focus to its prior location. Make sure forced focus changes, such as an `onChange` or `onfocus` event, are not used to change context.

5 Embedded Media and Object Elements

Programmatic objects that are embedded into a web page must be accessible. This means content such as Adobe Acrobat PDF documents, Flash movies, etc. must conform to the accessibility standards directly without an alternative. If accessible versions of the programs are not available, the embedded element should not be used. Additionally, synchronized captions are required for all audio and sounds in multimedia. Synchronized audio descriptions are required for all visual information including people's titles and visual actions that are important to the multimedia, and a transcript is not an acceptable solution to multimedia.

6 Animation and Dynamic Content

Dynamic content and animations may be distracting to users with cognitive impairments or low vision. Dynamic content or animation may cause assistive technology to read irrelevant information. Users with disabilities may not have sufficient time to read dynamically updating information or animated content. A mechanisms to stop, pause, or hide dynamic and animated content must be provided.

7 ARIA Roles, States, and Properties

ARIA defines attributes (role, states, and properties) to describe rich internet applications to assistive technology. When ARIA attributes (state, roles and properties) are not used correctly, assistive technology may not correctly function as expected. ARIA attributes must be used in conjunction with accessibility best practices such as keyboard access and other techniques to be successful.

8 Live Regions

Dynamically updating content that does not use an entire page as live regions. Areas of the page that allow multiple updates should be marked as live regions using the WAI-ARIA states and properties module. Live regions allow multiple updates to be announced by a screen reader without interrupting the user's

current focus in the content. Mark areas of the page that change as live regions using aria-live:

- "polite" for areas with normal precedence
- "assertive" for areas with urgent precedence
- Some ARIA role types are implied aria live regions (log, status, timer, alert, and marque)

9 Dialog Boxes

Focus moves to open dialog boxes and remains in the dialog box. Ensure dialog boxes use proper structure by indicating the boundaries of the dialog. Ensure that dialogs can be closed via the keyboard and keyboard focus returns properly from dialog boxes (such as to the control that activated the dialog).

10 Accessible Calendars

Pop-up calendar controls must be accessible via keyboard access. Programmatic indication of information such as selected date and important dates is also necessary. When calendar pop-ups are not accessible to keyboard only and users of assistive technology users may not be able to select a date or may enter the wrong date. A direct text entry for calendar controls is not an acceptable solution except where know the day or week or other information provided by a calendar is not needed.