Show & Hide: Accessible Patterns

Nick Hall

Senior Developer

Cordellia Yokum

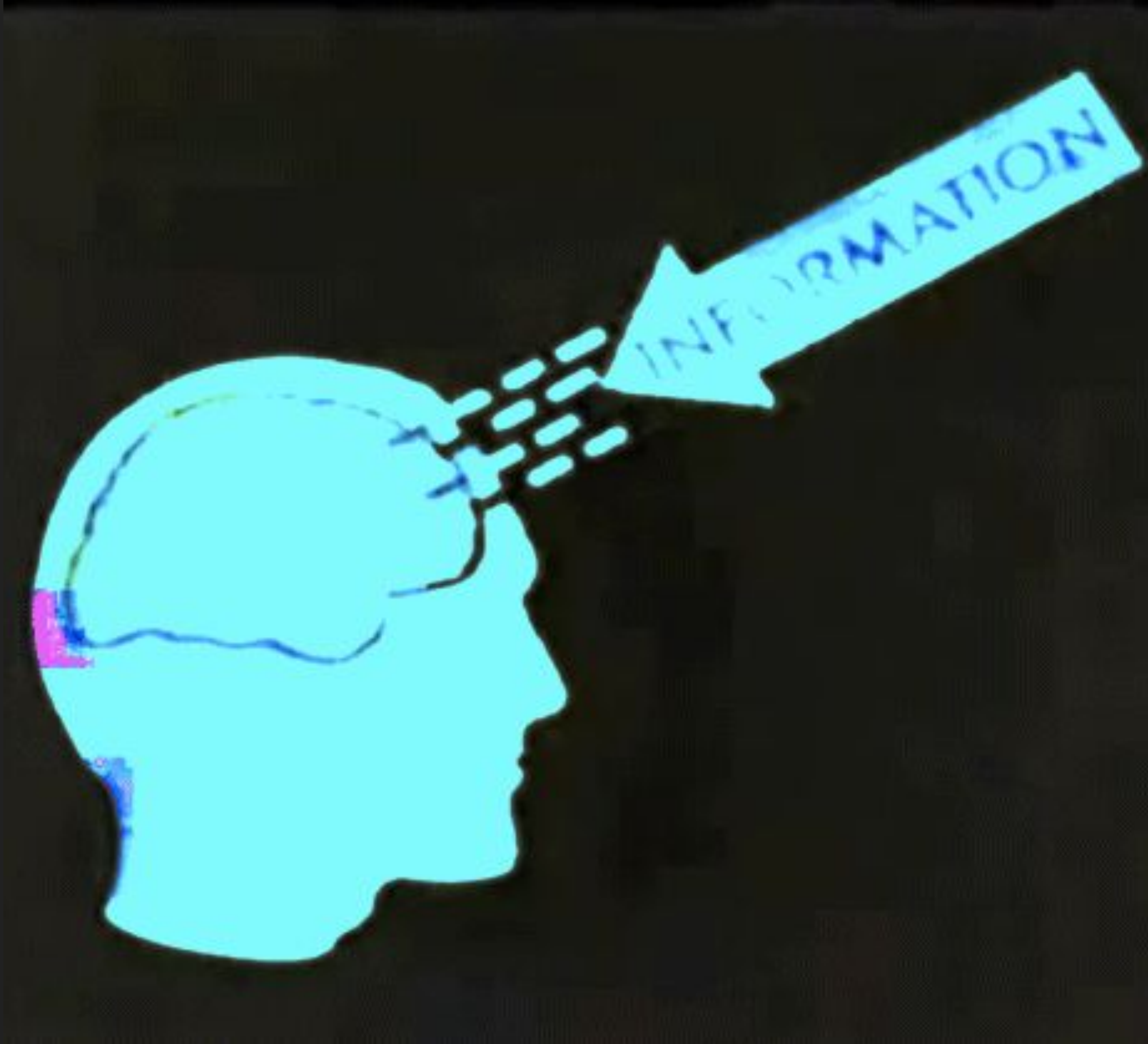
QA Manager

Animation & Motion

Designers ❤️ Motion

Draw Attention

Inform



Delight

And from an accessibility standpoint

R

Rules

WCAG 2.1

Interaction rules in 12 GIFs & one terrible screen cap

So, I'm incorporating emojis into my speech to better express myself. Winkie face.
#brooklyn9

Rules: Perceivable

1.3.1 Info and Relationships - Level A

Information, structure, and relationships conveyed through presentation can be programmatically determined or are available in text.

1.3.2 Meaningful Sequence - Level A

When the sequence in which content is presented affects its meaning, a correct reading sequence can be programmatically determined.

1.4.1 Use of Color
Level A

Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.

1.4.13 Content on Hover or Focus - Level AA

Where receiving and then removing pointer hover or keyboard focus triggers additional content to become visible and then hidden, the following are true:

Dismissable: A mechanism is available to dismiss the additional content without moving pointer hover or keyboard focus, unless the additional content communicates an input error or does not obscure or replace other content;

Hoverable: If pointer hover can trigger the additional content, then the pointer can be moved over the additional content without the additional content disappearing;

Persistent: The additional content remains visible until the hover or focus trigger is removed, the user dismisses it, or its information is no longer valid.

Rules: Operable

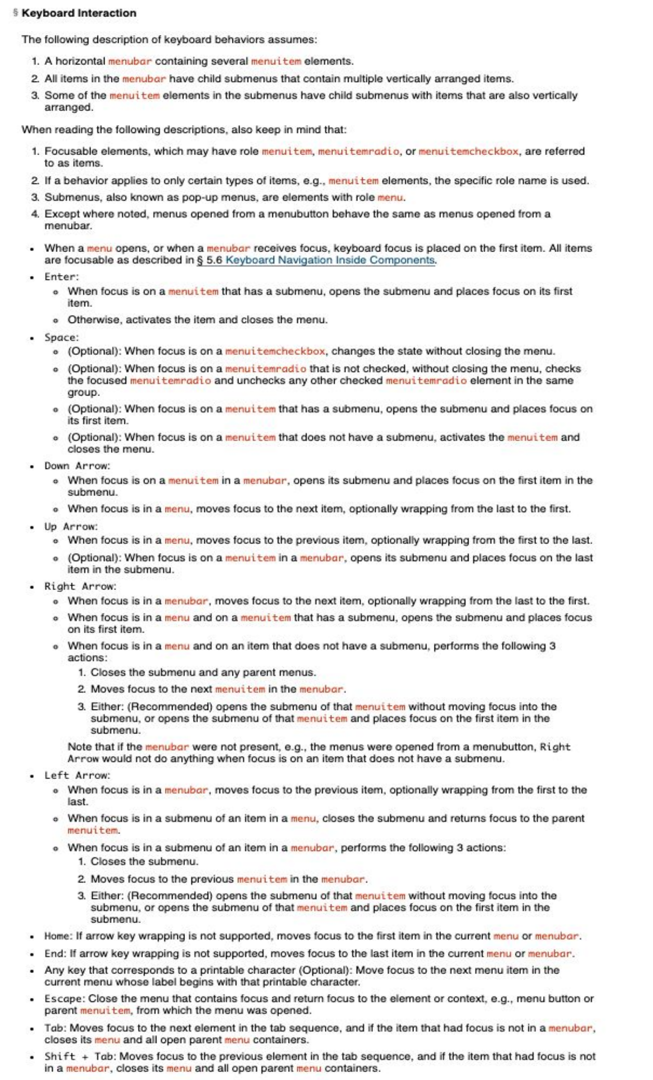
2.1.1 & 2.1.2
Keyboard - Level A

All functionality of the content is operable through a keyboard interface without requiring specific timings for individual keystrokes & focus can be moved away from that component using only a keyboard interface

42 Keyboard rules for just menu

Rules for menus. Including:

- Enter
- Space
- Down
- Up
- Right
- Left
- Home
- End
- Escape
- Tab
- Shift + Tab



§ Keyboard Interaction

The following description of keyboard behaviors assumes:

1. A horizontal `menubar` containing several `menuitem` elements.
2. All items in the `menubar` have child submenus that contain multiple vertically arranged items.
3. Some of the `menuitem` elements in the submenus have child submenus with items that are also vertically arranged.

When reading the following descriptions, also keep in mind that:

1. Focusable elements, which may have role `menuitem`, `menuitemradio`, or `menuitemcheckbox`, are referred to as items.
2. If a behavior applies to only certain types of items, e.g., `menuitem` elements, the specific role name is used.
3. Submenus, also known as pop-up menus, are elements with role `menu`.
4. Except where noted, menus opened from a menubutton behave the same as menus opened from a menubar.

- When a `menu` opens, or when a `menubar` receives focus, keyboard focus is placed on the first item. All items are focusable as described in § 5.6 Keyboard Navigation Inside Components.
- `Enter`:
  - When focus is on a `menuitem` that has a submenu, opens the submenu and places focus on its first item.
  - Otherwise, activates the item and closes the menu.
- `Space`:
  - (Optional): When focus is on a `menuitemcheckbox`, changes the state without closing the menu.
  - (Optional): When focus is on a `menuitemradio` that is not checked, without closing the menu, checks the focused `menuitemradio` and unchecks any other checked `menuitemradio` element in the same group.
  - (Optional): When focus is on a `menuitem` that has a submenu, opens the submenu and places focus on its first item.
  - (Optional): When focus is on a `menuitem` that does not have a submenu, activates the `menuitem` and closes the menu.
- `Down Arrow`:
  - When focus is on a `menuitem` in a `menubar`, opens its submenu and places focus on the first item in the submenu.
  - When focus is in a `menu`, moves focus to the next item, optionally wrapping from the last to the first.
- `Up Arrow`:
  - When focus is in a `menu`, moves focus to the previous item, optionally wrapping from the first to the last.
  - (Optional): When focus is on a `menuitem` in a `menubar`, opens its submenu and places focus on the last item in the submenu.
- `Right Arrow`:
  - When focus is in a `menubar`, moves focus to the next item, optionally wrapping from the last to the first.
  - When focus is in a `menu` and on a `menuitem` that has a submenu, opens the submenu and places focus on its first item.
  - When focus is in a `menu` and on an item that does not have a submenu, performs the following 3 actions:
    1. Closes the submenu and any parent menus.
    2. Moves focus to the next `menuitem` in the `menubar`.
    3. Either: (Recommended) opens the submenu of that `menuitem` without moving focus into the submenu, or opens the submenu of that `menuitem` and places focus on the first item in the submenu.

  Note that if the `menubar` were not present, e.g., the menus were opened from a menubutton, `Right Arrow` would not do anything when focus is on an item that does not have a submenu.
- `Left Arrow`:
  - When focus is in a `menubar`, moves focus to the previous item, optionally wrapping from the first to the last.
  - When focus is in a submenu of an item in a `menu`, closes the submenu and returns focus to the parent `menuitem`.
  - When focus is in a submenu of an item in a `menubar`, performs the following 3 actions:
    1. Closes the submenu.
    2. Moves focus to the previous `menuitem` in the `menubar`.
    3. Either: (Recommended) opens the submenu of that `menuitem` without moving focus into the submenu, or opens the submenu of that `menuitem` and places focus on the first item in the submenu.
- `Home`: If arrow key wrapping is not supported, moves focus to the first item in the current `menu` or `menubar`.
- `End`: If arrow key wrapping is not supported, moves focus to the last item in the current `menu` or `menubar`.
- Any key that corresponds to a printable character (Optional): Move focus to the next menu item in the current menu whose label begins with that printable character.
- `Escape`: Close the menu that contains focus and return focus to the element or context, e.g., menu button or parent `menuitem`, from which the menu was opened.
- `Tab`: Moves focus to the next element in the tab sequence, and if the item that had focus is not in a `menubar`, closes its `menu` and all open parent `menu` containers.
- `Shift + Tab`: Moves focus to the previous element in the tab sequence, and if the item that had focus is not in a `menubar`, closes its `menu` and all open parent `menu` containers.

2.2.2 Pause, Stop, Hide - Level A

There is a mechanism for the user to pause, stop, or hide:

Any moving, blinking or scrolling information that (1) starts automatically, (2) lasts more than five seconds, and (3) is presented in parallel with other content.

2.3.1 Three Flashes or Less Threshold - Level A

Web pages do not contain anything that flashes more than three times in any one second period, or the flash is below the general flash and red flash thresholds.

2.4.3 Focus Order - Level A

If a Web page can be navigated sequentially and the navigation sequences affect meaning or operation, focusable components receive focus in an order that preserves meaning and operability.

2.4.7 Focus Visible - Level AA

Any keyboard operable user interface has a mode of operation where the keyboard focus indicator is visible.

Rules: Understandable

3.2.4 Consistent Identification - Level AA

Components that have the same functionality within a set of Web pages are identified consistently.

Rules: Robust

4.1.2 Name, Role, Value Level A

For all user interface components states, properties, and values that can be set by the user can be programmatically set; and notification of changes to these items is available to user agents.

Patterns

Expanding Content

codepen

Meets Criterion

- Focus
- Keyboard
- Skip content
- Name, Role, Value (aria-pressed)

Modal

codepen

Meets Criterion

- Keyboard
- Name, Role, Value
  - Dialog
  - Aria-Hidden
  - Aria-Labelledby
  - Aria-modal
- Focus Order
- Focus Visible

Menu

codepen

Meets Criterion

- Keyboard
- Focus Order
- Focus Visible
- Name, Role, Value
    - Aria-labelledby
- Identify Menus <nav>
- Indicate Current Item

Accordion

codepen

Rules

- Focus
- Keyboard
- Skip content
- Name, Role, Value
    - Aria-labelledby
    - Aria-expanded

Hover Interaction

codepen

Rules

- Focus
- Keyboard
- Skip content
- Name, Role, Value
    -

Make it Fancy

Dean’s Financial Aid

Fancy Series of Modals

Beware

- Competing modals
- Focus traps
- Returning focus correctly

SPHS
Care Finder

Fancy Menus

Beware

- Labeling
- Keyboard Access
- Updated Content
- Mobile

UMass
Flex Callout

Fancy Hover Interaction & Expanding Content

Beware

- Labeling
- Focus
- Hoverable: Content does not disappear
- Consistent identification

Ford House Map

Fancy Expandable Content & Menu

Beware

- Mobile behavior
- Meaningful Sequence
- Focus visible only on the active item
- Returning focus correctly

In Conclusion

Think about interactions in its simplest elements

Meet criterion for that interaction

Build off of that.