What is the aria-expanded attribute, and how does it work?

The aria-expanded attribute is used for accessibility purposes to indicate if a control is expanded or collapsed. It's used in conjunction with collapsible widgets like menus, accordions and other disclosure widgets that control the visibility of content. The aria-expanded attribute is set to true if the control is expanded, or false if it is collapsed.

The information provided by aria-expanded allows people using screen readers to understand the current state of the control (whether it is expanded or collapsed).

The aria-expanded attribute is applied to the interactive element that toggles the visibility of a collapsible widget. For example, if a button toggles a expandable menu, the aria-expanded attribute is placed on the button.

When the menu is expanded, the aria-expanded attribute should be set to true like in this example:

<button aria-expanded="true">Menu</button>

When the menu is collapsed, it should be set to false instead.

<button aria-expanded="false">Menu</button>

You must always have the aria-expanded attribute set to either true or false on the controlling element. For example, if a button toggles the visibility of a menu, the default value for aria-expanded should be based on the default visibility of the menu.

If the menu is expanded by default, aria-expanded should initially be set to true. If the menu is collapsed by default, aria-expanded should initially be set of false.

The value of aria-expanded should be updated dynamically using JavaScript as the user interacts with the element.

Additionally, the properties, aria-controls and aria-owns can be used in combination with aria-expanded to establish a programmatic connection between the controlling element and the element it controls.

Let's start with aria-controls. When used with aria-expanded, the aria-controls attribute is used to specify the element being controlled. For example, a button might expand or collapse a list acting as a menu. The value of aria-controls will be the id of the controlled element (the menu list in this example).

<button aria-expanded="false" aria-controls="menu1">Menu</button>

<ul id="menu1">

<li>...</li>

<li>...</li>

</ul>

Notice that the list immediately follows the controlling button. For expandable controls like this, it is best to have the expanded content immediately follow the element that controls it in the DOM. This prevents screen reader users from having to search for the expanded content, and makes it easier for keyboard users to navigate through any interactive controls in the expanded content.

If it is not possible to place the expanded content immediately after the controlling element, the aria-owns attribute allows you to virtually move it after the control in the accessibility tree. This allows assistive technology like screen readers to pretend the expanded content is placed directly after the control in the DOM.

<button aria-owns="menu1" aria-expanded="true">Menu</button>

<main>

<!-- an entire page worth of content --->

</main>

<ul id="menu1">

<li>...</li>

<li>...</li>

</ul>

There are drawbacks to using the aria-owns attribute. It creates unnecessary verbosity for screen reader users since most screen readers will automatically read out the entire contents of the expanded element when first expanded. It also does not change the tab order, potentially forcing keyboard users to tab through all of the other interactive controls on the page before reaching the expanded content, unless you manage the tabbing order with JavaScript.

Ideally, the expandable content should be placed after the control element, and the aria-owns attribute should only be used in a worst case scenario when that is not possible. If it must be used, you will need to thoroughly test with a wide range of screen readers and browsers to ensure that your implementation is accessible for everyone.

And just a reminder, when you use either aria-controls or aria-owns, the value of aria-expanded must continue to be updated as the control is expanded and collapsed.

The aria-expanded attribute indicates whether a control is expanded or collapsed. This information is essential for screen reader users, helping them understand the current state of collapsible elements like menus, accordions, and other similar disclosure widgets.

By using aria-expanded correctly, you can create an intuitive user experience for everyone.