HowTo Components – Overview

- Components
- Goals
  - Accessibility
  - Performance
  - Visual Flexibility
  - Maintainable code
- Non-Goals
  - Be a library / framework / toolkit
  - Be backwards compatible
  - Be complete

"HowTo: Components" are a collection of web components that implement common UI patterns. The purpose of these implementations is to be an educational resource. You can read through the densely commented implementation of different components and hopefully learn from them. Note that they are explicitly **NOT** a UI library and should **NOT** be used in production.

Components

- `<howto-checkbox>`: represents a boolean option in a form. The most common type of checkbox is a dual-type which allows the user to toggle between two choices—checked and unchecked.
- `<howto-tabs>`: limits visible content by separating it into multiple panels.
- `<howto-tooltip>`: a popup that displays information related to an element when the element receives keyboard focus or the mouse hovers over it.

Goals

Our aim is to demonstrate best practices for writing robust components that are accessible, performant, maintainable, and easy to style. Each component is completely self-contained so it can serve as a reference implementation.

Accessibility

The components closely follow the WAI ARIA Authoring Practices, which is a guide to explain and show ARIA, the Accessible Rich Internet Application standard. If you are unfamiliar with ARIA, check out our introduction on WebFundamentals. Each component links to the relevant section of the Authoring Practices. While not strictly necessary, we do recommend reading the section of the Authoring Practices before diving into the code.

Performance

In web development the term "performance" can be applied to a multitude of things. In the context of `<howto>`, performance mostly refers to animations consistently running at 60fps, even on mobile devices.

Visual Flexibility

As much as possible, components are not styled, except for layout or to indicate a selected or active state. This is to keep the implementation visually flexible and focused. By not spending time on decoration, we limit the code to only what is absolutely necessary to make the component function. If any style is required for the component to function, the style will be marked with a comment explaining why that is.

Maintainable code

As HowTo: Components is a reference implementation, we spent extra time on writing readable and easily comprehensible code that is densely commented.

Non-Goals

Be a library / framework / toolkit

`<howto>` components are not published on npm, bower or any other platform because they are not meant to be used in production. For the sake of terse, readable code, we are using modern JavaScript APIs and are supporting modern browsers which implement the Web Components standards. You will be able to adapt the code to fit your own needs after reading these implementations.

Be backwards compatible

The code should not be relied on directly. We might, and very likely *will*, drastically change the implementation and API of any element if a better implementation is discovered. This is a living resource where we can share, explore, and discuss best practices for building web UIs.

Be complete

We currently don't (and probably won't) implement *_all_ *components that can be found in the WAI ARIA Authoring Practices. However, re-using the principles used in other `<howto>` components should enable readers to implement any components that are missing.

□□□ #2
□ □□□ □□□ 2026-10-31 21:45:41 UTC
□ □□□ □□□ 2026-10-31 21:45:42 UTC