Setting Up an Accessible Web Page: Making Buttons Accessible (A UX Accessibility Blog Post)

This UX Accessibility Blog Series, “Setting Up An Accessible Web Page,” will walk you through the basic elements of an accessible web page in hopes that you can take what you’ve learned an apply it to more complex pages. This blog series is loosely based on Heydon Pickering’s book [“Inclusive Design Patterns: Coding Accessibility Into Web Design.”](https://www.smashingmagazine.com/inclusive-design-patterns/)

We’ve talked about what needs to be done to make buttons accessible multiple times before, but since the relationship between these elements and accessibility/inclusivity is an important conversation to have, we’ll go over them (maybe in a bit more detail) again.

**General Button Accessibility Tips**

As always, using semantic HTML is always preferable when it comes to accessibility. Most browsers know how to accessibly deal with semantic mark-up by default. Native HTML buttons (<button>, <input type=”button” />, <input type=”submit” />, <input type=”reset” />) also support keyboard and focus requirements without needing any other customization, according to Mozilla Developer Network’s “Using the button role.”

Button tags must never be left empty. If your button contains only a glyphicon, make sure you make the glyphicon button accessible by adding a <span class=”sr-only”>[Action glyphicon represents]</span> after your glyphicon. Assistive technologies can’t read glyphicons, so they will understand the button as empty. This is why if you run WAVE or a similar accessibility checker on a page that has a glyphicon button, it may display an error stating the button is empty.

**Colors**

Color can be used in a way that quickly and efficiently tells the user what a button is used for. It is best if the color of the buttons follow convention, as shown in Ensemble. Our users are acclimated (especially those that use our newer systems) to the three main button colors, blue, black, and grey. Each color corresponds with a particular type of action. For submit buttons, we typically use blue (.btn-primary) buttons. Our users, and most other users, associate this with an action like submitting, saving, etc. For delete buttons, we use black (.btn-inverse) buttons. For any action that doesn’t submit, save, or delete, we use grey (.btn-default) buttons. The grey buttons are typically for cancelling an action, resetting a form, etc.

**Buttons as Links**

If you have an anchor tag that is styled like a button and redirects to another page, do not add the role=”button” attribute. Adding role=”button” will confuse your users since they are expecting an action to take place on that page.

Typically, modals won’t trigger correctly if you use a <button> element. In this case, it’s necessary to use <a href=”javascript:void(0)” role=”button” class=”btn btn-default”>Button</a>. In this example, using role=”button” is good since it triggers an event on the page and doesn’t redirect the user to a different page. As Bootstrap framework developers work to make elements more accessible, this may change. But for now, the downside to this is semantics. Assistive technologies will see the anchor tag as a link before they’ll see it as a button. It’s ALWAYS better to use semantic HTML.