Setting Up an Accessible Web Page: Skip Links and The Main Element (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/)

**<a href=”#main” class=”sr-only sr-only-focusable”>Skip to Main Content</a>**

Screen readers start out by first, as we discussed in the last blog post, announcing the page title. The next thing it starts to announce is the navigation bar. If this isn’t the first page the user has visited on your site, they may not want to hear the navigation spiel over and over again. It will also help keyboard users to not tab through every link in your navigation to get to where they need to go in your main content. “Skip to Content” links can help your users bypass blocks of repeated content and generally have a better experience on your website.

It is important for this skip to content link to be the first element in your body tag for it to be beneficial. These skip links shouldn’t be visible by default because a lot of your users won’t need it (and it’s kind of an eyesore). To show your skip content link to those who actually need it, just add class=”sr-only sr-only-focusable”. This ensures that screen readers will read the skip content link immediately after announcing the page title and will be the first thing your keyboard users can navigate to.

**<main id=”main”>**

The main element is used to contain the main content of the page (of course). This content will be unique from page to page. According to the W3C HTML Editors Draft, the main element “consists of content that is directly related to or expands upon the central topic of a document or central functionality of an application.” It was introduced to HTML5 formally as a way to map the ARIA role “main” to an HTML element, (ideally) eventually phasing out the need for role=”main”.\* Many screen readers and other assistive technologies recognize the main tag as where the main content of the page begins and gives the user a way to bypass repetitive content, such as navigation bars.

Since the main element represents the main content, it should be used only once per web page. It also can’t be used as a child element of any other document outline element, such as <article>, <aside>, <footer>, <header>, and <nav>. Also, it is important to make sure your navigation bar is not included in your <main> element since it is repeated on all of the pages of your website.

Some people still print out web pages. This is very possible since we work with clients that sometimes need to go out in the field and have the information we provide to them on hand. The main element also makes it easier for us to accommodate users who print out our web pages. The main element helps these users by targeting an area of our page that has all of the information they need to be able to print (i.e. data tables) and none of the stuff they don’t (i.e. navigation elements).

\*NOTE: You should still use role=”main” until all browsers map the role to <main> automatically. I believe IE11 and earlier still don’t. In older versions of IE, it may be necessary to add <script>document.createElement(‘main’);</script> since it doesn’t recognize main as an element.

**</main>**