# Notes on Front End Web Development: Things beyond the basics that developers should understand

Note: hyperlinks from word docs don’t always play nice with cmich.edu login and may need to be copied and pasted into the address bar.

## Accessibility

There are two main sets of guidelines for this, and they for the most part overlap.

* **ADA 508:** <https://www.ada.gov/508/>  
  Americans with Disabilities Act Section 508 deals with the web. It is the standard that government entities’ websites are legally held to. CMU is somewhat of a government entity.
* **WCAG 2.1:** <https://www.w3.org/TR/WCAG21/>Web Content Accessibility Guidelines set by W3C (World Wide Web Consortium). CMU has a goal of meeting WCAG 2.1 level AA compliance.

Before reading that nitty gritty stuff though, you might want to start here for a more high-level view:

**Making Web Content Accessible**: <https://cmich.teamdynamix.com/TDClient/KB/ArticleDet?ID=35849>

**Digital Content Accessibility Checklist**: <https://www.cmich.edu/office_provost/CIS/Pages/Explore%20Teaching%20and%20Learning/Selecting%20or%20Developing%20Materials%20and%20Tools/Accessibility-Checklist.aspx>

## Branding and Identity

Like any major institution, CMU has a Style Guide document (also known as a Master Brand Palette) to define standards for Font types, Colors, icons, etc. A wealth of information on this can be found here:  
<https://www.cmich.edu/office_president/university_communications/secure/Pages/Identity-Standards.aspx>

## W3C Validator

W3C Validator is a good first step to make sure your website doesn’t have a broken DOM from things such as elements with duplicate IDs, unclosed tags, and elements wrapped in other elements they don’t belong inside of (for example, an inline element such as a <span> wrapping a block level element such as <div> or <button>, which is the HTML coding equivalent of wearing underpants on the outside).

It’s a good practice to make sure pages pass this test without errors, but sometimes you get stuck having to use a Framework (SharePoint comes to mind) that will have unavoidable HTML errors. One thing to know about W3C validator is that when validating via URI, it only sees the page the way it’s initially loaded. DOM manipulation that happens after the initial page load is not seen by this unless you paste the code into it directly (for example, by opening dev tools and getting the *outer* HTML of the <HTML> tag after the page *completely* loads). Feel free to give it a whirl at <https://validator.w3.org/>

## Monsido

Monsido is another validator tool that is similar to W3C validator above. It does a lot more, but it’s not free. None the less, this is what CMU uses. It will flag many different types of errors, including accessibility. However, this is still no substitute for actually trying to use a website blindfolded.

## Screen Readers

Screen readers are part of a family of programs known as **Assistive Technologies**. They allow those that are less than fully able to see what’s on screen to navigate a website by sound. Two of the better screen readers for Windows operating systems are Microsoft Narrator (which is built into windows), and NVDA, which is open source and supported by community efforts. On Mac OS and iOS, VoiceOver is the name of the built-in screen reader. Android has a screen reader called TalkBack. Both mobile device screen readers allow navigation with swipes and taps.   
  
With any of these screen readers, you will find that often a website may technically pass most ADA and WCAG requirements, but still be off-putting to those who depend on a screen reader. Learning to navigate by keyboard alone while using a screen reader is a skill set may web developers never even consider, yet it’s an important way to find inaccessible parts of websites, as well as “keyboard traps” that testing software can sometimes overlook.

To download NVDA for windows: <https://www.nvaccess.org/download/>

## Other good things to know

* CSS
  + Box model (Border box is your friend)
  + Display: Inline vs Block
  + Float and Clear
  + Positioning: Absolute, Relative, and Fixed
  + Print CSS (this one is very overlooked!)
  + Visually Hidden CSS class to help control screen readers (SharePoint uses “.sr-only”)
  + Specificity vs !important (try not to abuse !important)
* jQuery/JavaScript
  + CSS selectors
  + Animations (hint: CSS 3 transitions are preferred to Dom manipulation for performance)
  + DOM traversal
* Bootstrap Library
  + 12 Column Grid system
  + CSS media Queries
  + Hidden/Visible classes (hidden-xs vs visible-xs in Bootstrap 3 for example.)