## Criterion E: Product development

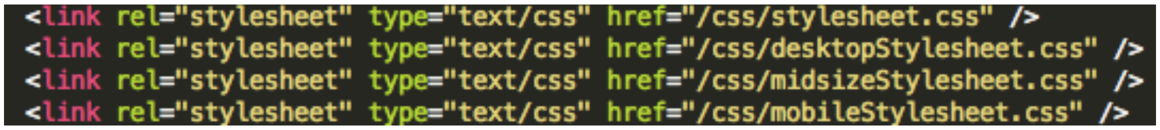
Complex Techniques used to Satisfy Client's Requirements

- Use of Cascading Style Sheets to customize visual appearance of website
- Use of PHP to auto-generate content
- Use of JavaScript to implement button press functionality

## Use of Cascading Style Sheets (CSS)

CSS stands for Cascading Style Sheets, and is a markup language used to specify the visual styling one wants to apply to the html of a website.

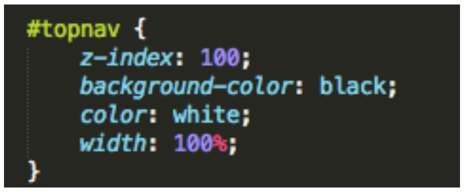
The CSS used within these pages was written manually into a text editor (Sublime Text). Almost all pages in the site contain these links:



```
<link rel="stylesheet" type="text/css" href="/css/stylesheet.css" />
<link rel="stylesheet" type="text/css" href="/css/desktopStylesheet.css" />
<link rel="stylesheet" type="text/css" href="/css/midsizeStylesheet.css" />
<link rel="stylesheet" type="text/css" href="/css/mobileStylesheet.css" />
```

These links each lead to a CSS resource. The rel="stylesheet" part specifies that the files being linked are style sheet files, the type="text/css" states the content of the files are text based and that they are .css files, and the href="something" parts specify the location of the files being linked.

Note that there is a "/" at the beginning of the URLs within the href attribute. The "/" specifies that the path followed should start from the root of the file system (and are relative to the root of the file system, rather than the file this series of <link>s are declared in). Because of this, generally these links will not work if they are run locally, unless the root of the website is placed in the root of the computer's file system. However, this does work when the site is uploaded, and has the benefit of making my <link> tags consistent throughout the site, reducing the need to refactor each page (this same technique is used with the header).



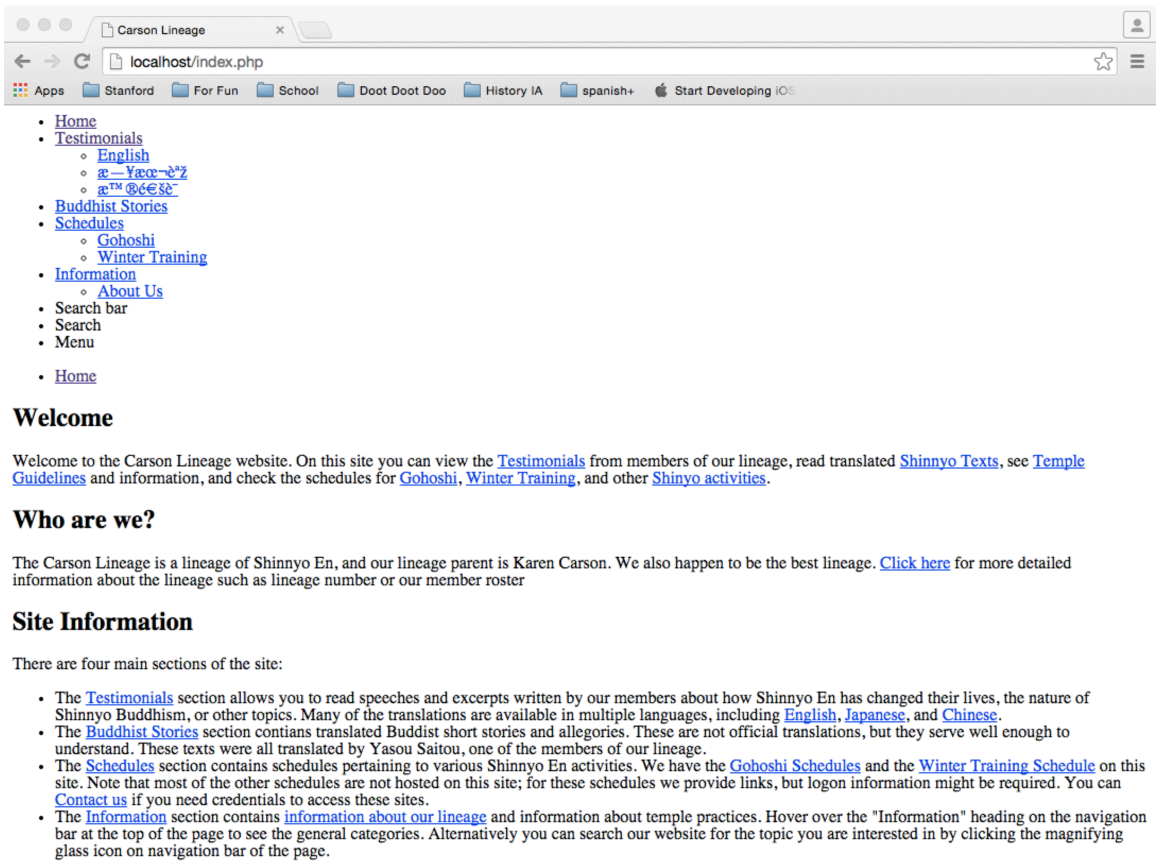
```
#topnav {
    z-index: 100;
    background-color: black;
    color: white;
    width: 100%;
}
```

The actual CSS style sheets look like the image to the left. The block shown selects all elements with the ID "topnav" (which happens to be a <div> that is our navigation bar). This block sets visual attributes of our navigation bar.

Specifically, what we are doing is setting the z-index (the precedence of 'height' on the screen) to 100, we set the background color to black, we set the text color to white, and we set the width of the navigation bar to be 100% of the width of the element that is around it (which is the <body> element).

Of course, this is not the only CSS styling that I apply; overall, I apply CSS styling to over 16 elements, creating a visually appealing website. Examples are on the next page.

Before CSS:



- Home
- Testimonials
  - English
  - æ—¥æœ¬èªž
  - æ™®é€šè¯
- Buddhist Stories
- Schedules
  - Gohoshi
  - Winter Training
- Information
  - About Us
- Search bar
- Search
- Menu

- Home

**Welcome**

Welcome to the Carson Lineage website. On this site you can view the Testimonials from members of our lineage, read translated Shinnyo Texts, see Temple Guidelines and information, and check the schedules for Gohoshi, Winter Training, and other Shinyo activities.

**Who are we?**

The Carson Lineage is a lineage of Shinnyo En, and our lineage parent is Karen Carson. We also happen to be the best lineage. Click here for more detailed information about the lineage such as lineage number or our member roster

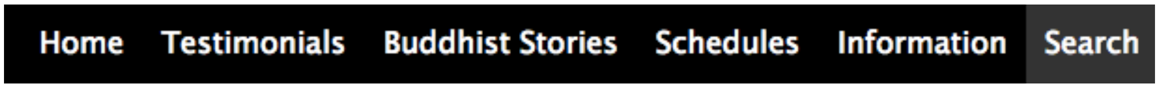
**Site Information**

There are four main sections of the site:

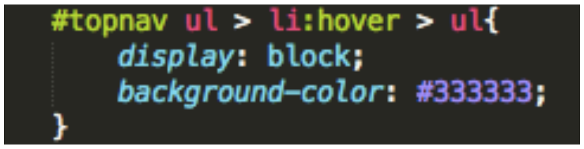
- The Testimonials section allows you to read speeches and excerpts written by our members about how Shinnyo En has changed their lives, the nature of Shinnyo Buddhism, or other topics. Many of the translations are available in multiple languages, including English, Japanese, and Chinese.
- The Buddhist Stories section contians translated Buddist short stories and allegories. These are not official translations, but they serve well enough to understand. These texts were all translated by Yasou Saitou, one of the members of our lineage.
- The Schedules section contains schedules pertaining to various Shinnyo En activities. We have the Gohoshi Schedules and the Winter Training Schedule on this site. Note that most of the other schedules are not hosted on this site; for these schedules we provide links, but logon information might be required. You can Contact us if you need credentials to access these sites.
- The Information section contains information about our lineage and information about temple practices. Hover over the "Information" heading on the navigation bar at the top of the page to see the general categories. Alternatively you can search our website for the topic you are interested in by clicking the magnifying glass icon on navigation bar of the page.

Between this shot and the next I changed the styling on quite a few elements.

For the list of links at the very top, I changed it so that the second level of bullets would be hidden, and so that the first level of links would align horizontally rather than moving out vertically. This had the effect of making a menu bar that looks like this:

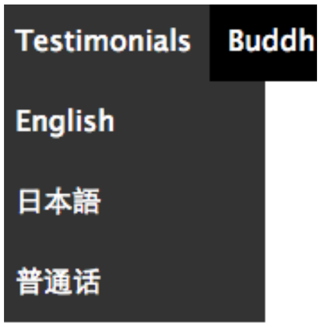


Secondly, I used the CSS pseudo-class :hover to select the sub-bullets of menu buttons that I hovered over and make them visible with this code (notice the li:hover selects the hovered-over menu items):

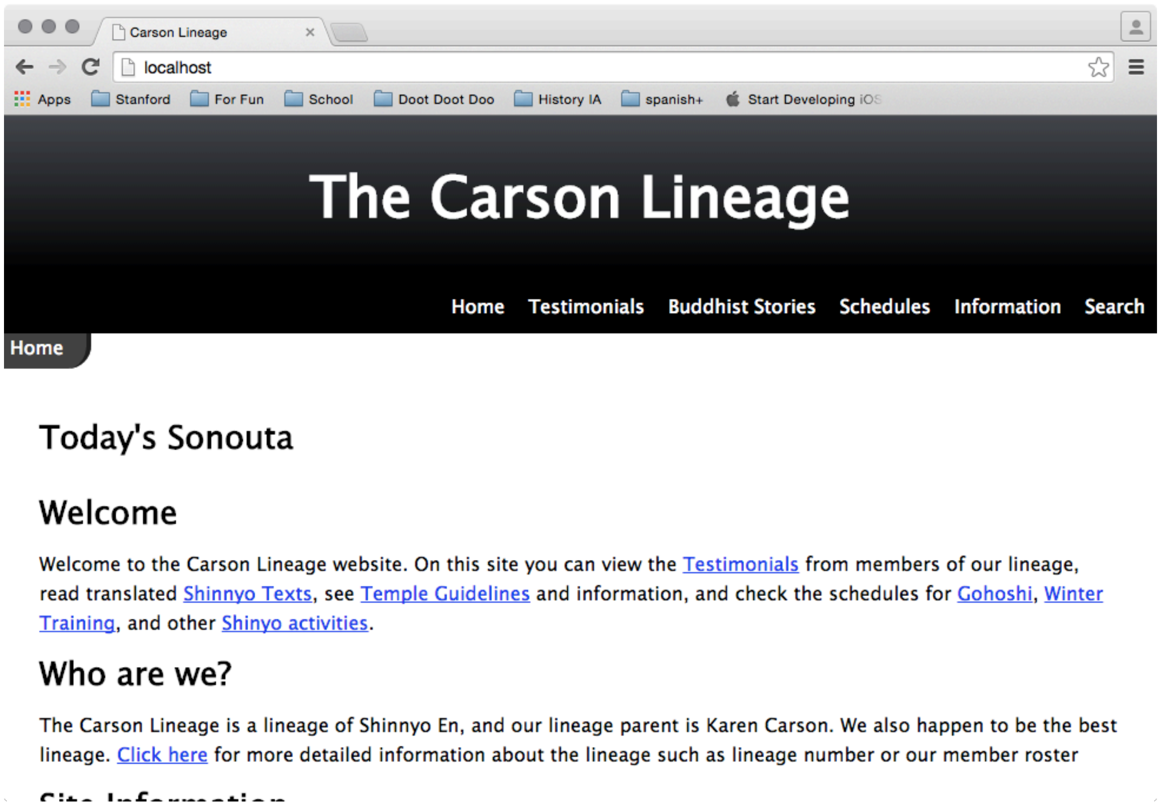


```
#topnav ul > li:hover > ul{
    display: block;
    background-color: #333333;
}
```

Which produces an effect like this when some of the menu items are hovered over:



I applied less detailed visual changes to the rest of the site's elements, and so the page now looks like this after CSS:



The Carson Lineage

Today's Sonouta

Welcome

Welcome to the Carson Lineage website. On this site you can view the Testimonials from members of our lineage, read translated Shinnyo Texts, see Temple Guidelines and information, and check the schedules for Gohoshi, Winter Training, and other Shinyo activities.

Who are we?

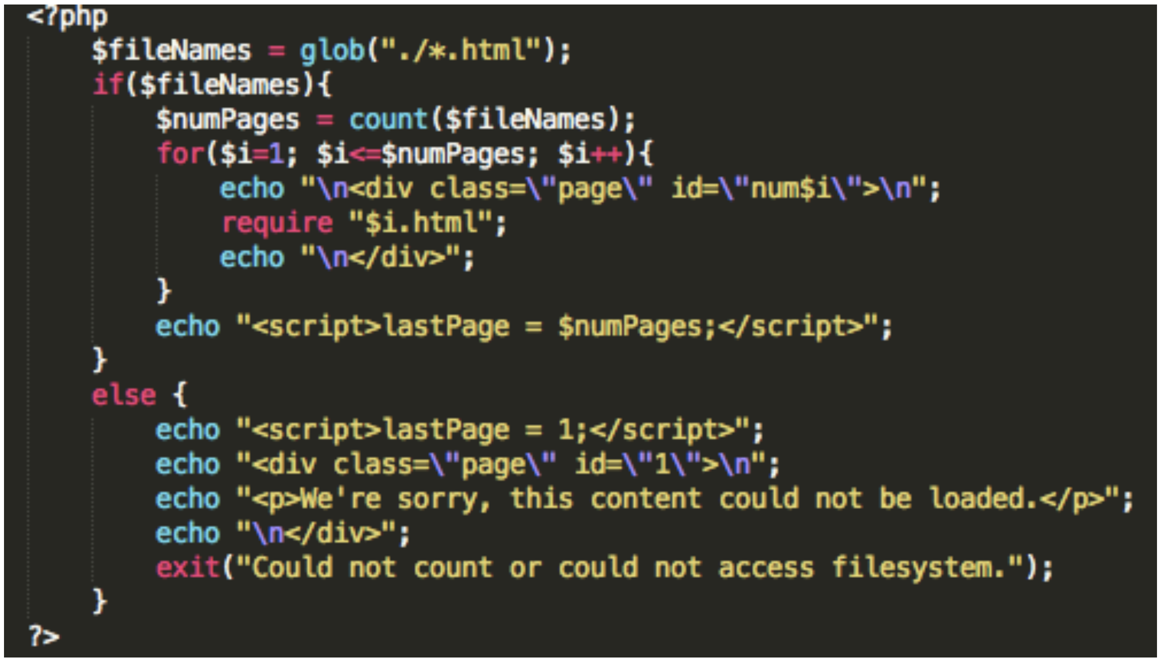
The Carson Lineage is a lineage of Shinnyo En, and our lineage parent is Karen Carson. We also happen to be the best lineage. Click here for more detailed information about the lineage such as lineage number or our member roster

**Use of PHP to Augment Scalability and Auto-generate Content**

PHP: Hypertext Processor is a scripting language that processes HTML on the server before serving it to someone requesting the webpage.

The site has a lot of "reader pages" which display the testimonials and Buddhist stories; however, I do not want to mix the html for the reader itself and the plaintext content that I have, because that hurts scalability and I would have a hard time changing it later.

In order to avoid this I use this PHP code



```
<?php
    $fileNames = glob("./*.html");
    if($fileNames){
        $numPages = count($fileNames);
        for($i=1; $i<=$numPages; $i++){
            echo "\n<div class=\"page\" id=\"num$i\">\n";
            require "$i.html";
            echo "\n</div>";
        }
        echo "<script>lastPage = $numPages;</script>";
    }
    else {
        echo "<script>lastPage = 1;</script>";
        echo "<div class=\"page\" id=\"1\">\n";
        echo "<p>We're sorry, this content could not be loaded.</p>";
        echo "\n</div>";
        exit("Could not count or could not access filesystem.");
    }
?>
```

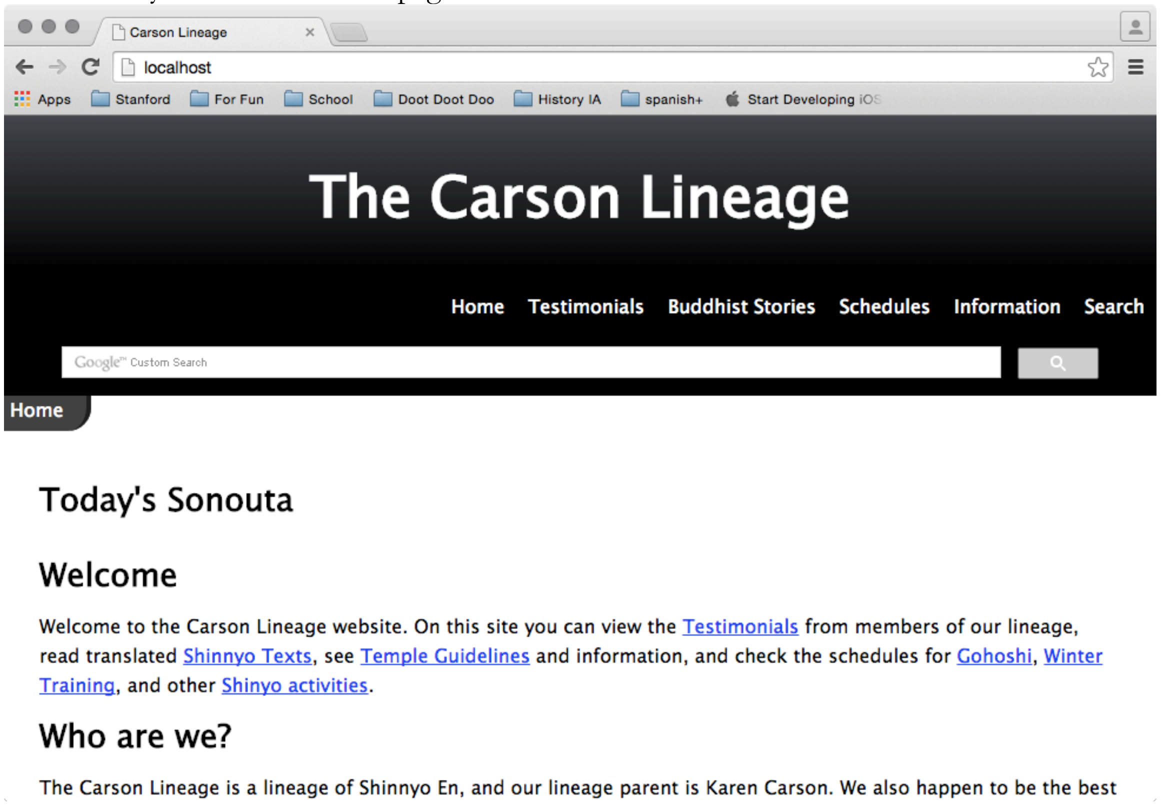
This counts the number of .html files are in the same directory as the reader index page, and it coalates all of them into separate <div>s on the reader page so that they may be displayed within the reader page. Glob() returns an array of the filenames that match the given regular expression, I then count the number of them and import each one of them inside of a div using the 'echo' and 'require' methods. The else statement at the end of this statement is just error handling code.

Using this code means that I can separate my reader view from my actual content, meaning that, in the future, I can just replace the reader page rather than having to extract the content itself all over again.

**Use of JavaScript**

JavaScript is a programming language common among most major browsers; the language is interpreted and executed on the clients machine, allowing one to update or modify a page quickly on a user's machine without using up server resources.

In this project I use JavaScript in order to implement the functionality of buttons on certain pages. For example, I would like my site to have a search bar, but it is rather inelegant to have it always be shown on the page like so:



Today's Sonouta

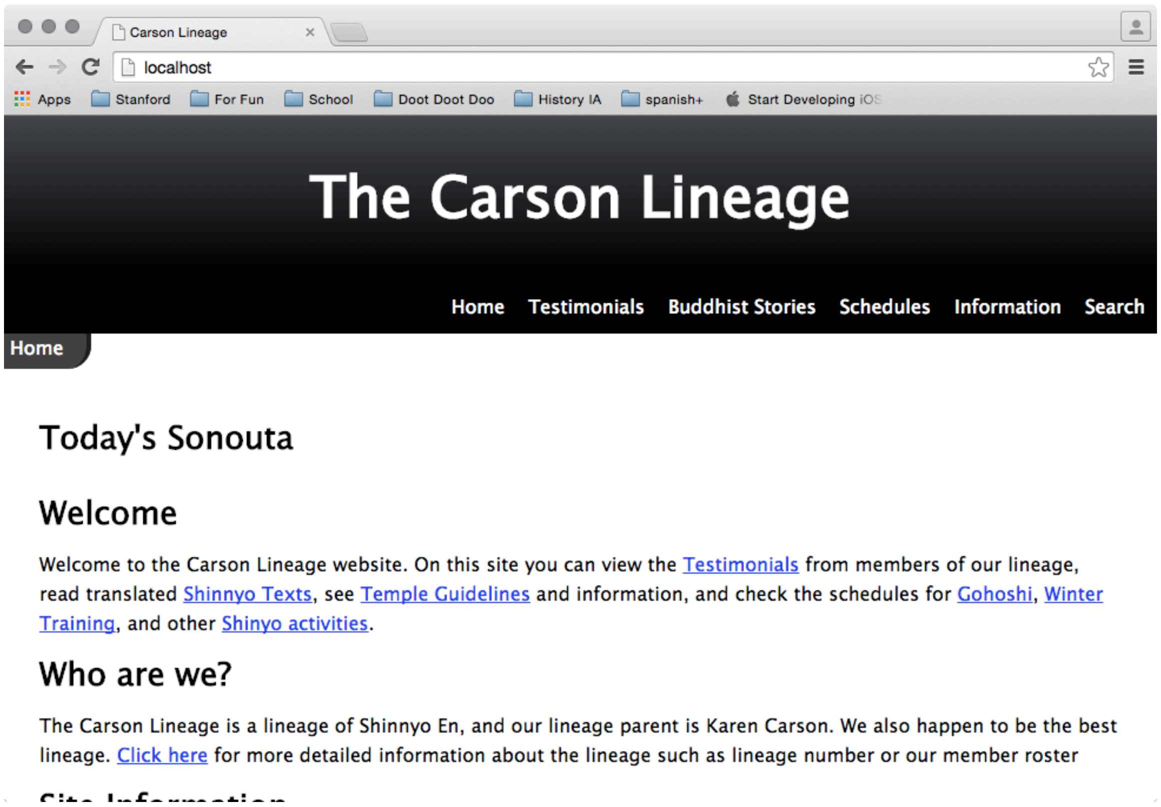
Welcome

Welcome to the Carson Lineage website. On this site you can view the Testimonials from members of our lineage, read translated Shinnyo Texts, see Temple Guidelines and information, and check the schedules for Gohoshi, Winter Training, and other Shinyo activities.

Who are we?

The Carson Lineage is a lineage of Shinnyo En, and our lineage parent is Karen Carson. We also happen to be the best

Especially on mobile where vertical horizontal space are already at a premium. Therefore, I have created a search button. By clicking the "Search" text (which is to be replaced by a magnifying glass icon later) you can toggle whether the search bar is shown, resulting in this:



The Carson Lineage

Today's Sonouta

Welcome

Welcome to the Carson Lineage website. On this site you can view the Testimonials from members of our lineage, read translated Shinnyo Texts, see Temple Guidelines and information, and check the schedules for Gohoshi, Winter Training, and other Shinyo activities.

Who are we?

The Carson Lineage is a lineage of Shinnyo En, and our lineage parent is Karen Carson. We also happen to be the best lineage. Click here for more detailed information about the lineage such as lineage number or our member roster

This is done using a JavaScript script loaded within the header like so:

```
<script src="/js/navbarButtons.js"></script>
```

This links to a script which executes whenever the page load (we do this by setting the script to run on window.onload):

```
1  window.onload = function() {
2
3      var searchButton = document.getElementById('searchbutton');
4      searchButton.onclick = function() {
5          var searchBar = document.getElementById('searchbar');
6          if(searchbar.style.display == "none"){
7              searchbar.style.display = "block";
8          }
9          else{
10             searchbar.style.display = "none";
11         }
12     }
13 }
```

In this script the window.onload is set to a wrapper function which holds the code which gives the search button its functionality. On line 3 we find the searchButton which we are adding the functionality to. Note that the JQuery library is normally used for things like this; however, the script and page are so light that it just isn't worth loading an extra script. On line 4 we again use a wrapper function in order to set the functionality of the searchButton. The rest of the lines use an if-statement in order to toggle the hidden state of the search bar.

I am going to implement something similar with a "menu" button to conditionally show the navigational components of the navigation bar.

Word Count: 1043