**Lesson 9: Navigation Bars**

**The Set Up:**

***Preparation:***

Read through the information below, taking time to click on any links to W3chools tutorials, and experimenting with their code. Once you have read through and understood the information below, complete “The Doing” activity at the end of this lesson, then hand it in to Ms. Hammond. Don’t forget to consider the comprehension questions!

**The Learning:**

***Navigation Bars***

A navigation (nav) bar is a stylized (with CSS) list <ul> of links <a> that appears either horizontally (across the top of your webpage) or vertically (along the left or right side). There are three parts to creating a Navigation Bar:

1. Create an unordered list of links you want in your Nav bar
2. Use CSS to style this list of links to make it LOOK like a Nav bar
3. Decide whether you want a vertical or horizontal Nav Bar, then format it accordingly.

***Creating Navigation Bars -*** *Experiment with W3School’s* [*Navigation Bars Tutorial*](https://www.w3schools.com/css/css_navbar.asp)

Step 1: Create an unordered list of links:

1. In your HTML document, create a new set of <div></div> tags. Inside the start tag, define a class that you would associate with a navigation bar (example: class=“nav”).
2. Nested between the <div> tags, create an unordered list using the <ul></ul> tags to define the list and the <li></li> tags to define each separate list item
3. Inside each set of <li></li> tags, nest a set <a href=“url”></a> tags containing the URL of each link you want to make. Make sure that you type the link name (the name of the link you want to be displayed in the Nav bar) between the <a></a> tags.

<ul>

<li><a href="#">Home</a></li>

<li><a href="#”>News</a></li>

<li><a href="#">Contact</a></li>

<li><a href="#">About</a></li>

</ul>

Step 2: Style the list of links like a nav bar:

1. First, you’ll need to make some changes to the way the unordered list, itself, it formatted. Remove the bullets from the list by using the “list-style-type” property. Remove both margins and padding from all the links so that there is no space between them to form a continuous bar. In your CSS, use the **“.nav ul”** (entire unordered list within the “nav” div) element as your selector, and add the following CSS Rules:
   * list-style-type: none;
   * margin: 0px;
   * padding: 0px;
2. Next, you need to make some changes to the way the links are formatted. Remove the underline from under each link by using the “text-decoration” property. In your CSS, use the **“.nav ul li a”** (individual list item > link within the “nav” div) element as your selector, and add the following CSS Rule:
   * text-decoration: none;

Step 3: Position your list, either vertically or horizontally:

1. Decide whether you would like to create a vertical or horizontal nav bar for your website, then follow the directions in the appropriate section, below:

***Vertical Navigation Bars***

1. To create a vertical nav bar, you’ll need to make some changes to the way the list items are formatted. Change the display to block so that each link’s entire box (rather than just the text) becomes editable and clickable. Adjust the width to however wide you want the vertical nav bar to appear across the screen (note: you can also set the width of the “ul” element, instead of the “li” element). Change the background color so that you can see the changes you’ve made, and so that the nav bar stands apart from the rest of the page. In your CSS, use the **“.nav ul li a”** (individual list item > link within the “nav” div) element as your selector and add the following CSS Rules:
   * display: block;
   * width: 60px;
   * background-color: mediumturquoise;
2. Make your nav bar more visually appealing by adjusting the padding between your links, the alignment of your links, and the color of the text within those links! In your CSS, use the **“.nav ul li a”** (individual list item > link within the “nav” div) element as your selector and add the following CSS Rules:
   * color: white;
   * padding: 10px;
   * text-align:center;
3. Add pseudo-classes to create more user interactivity and special effects [if this is your first time reading this information, see the “Styling Link States,” “Pseudo-classes,” and “pseudo-elements” sections further down in this document before completing this step]. In your CSS add and style pseudo-classes that will alter the “visited,” “hover,” and “active” states of the link, **in that order**:
   * .nav ul li a:visited {color:black;}
   * .nav ul li a:hover {background-color:darkcyan;}
   * .nav ul li a:active {font-weight:bold;}

Tip/Trick: To make your navigation buttons more defined, you can add borders around the entire “ul” element and/or individual “li” items.

Extension – [Sticky Vertical Nav Bar](http://www.w3schools.com/css/css_navbar.asp): Click the link to s W3School’s tutorial

***Horizontal Navigation Bars***

1. To create a horizontal nav bar, you’ll need to make some changes to the way the list items are formatted. To make a quick, simple horizontal nav bar, you can simply use the “.nav li” (individual list item within the “nav” div) element as your selector, then change the “display” property to “inline” (the default is “block”). This will cause the individual links to display on the same line as one another:
   * .nav li {display: inline;}

**OR**

1. To make a more dynamic horizontal nav bar, you can adjust the positioning, display, and background color. Use the **“.nav ul li a”** (individual list item > link within the “nav” div) element as your selector, then change the “float” property to “left.” This will cause the individual links to slide next to one another on the same line. You can use the “clear” property, combined with the “left” value on any elements that happen to be below the nav bar. This will clear everything to the left of these elements. Change the “display” to “block” so that each link’s entire box (rather than just the text) becomes editable and clickable.
   * float: left;
     + Note: Whenever you float elements, it collapses their “css box” – thus, if you want to change the background color/image, you must specify a width and height.
   * display: block;
2. Change the background color so that you can see the changes you’ve made, and so that the nav bar stands apart from the rest of the page. Use the “overflow:hidden” property to ensure that the background color fills only the unordered list. In your CSS, use the **“.nav ul”** (entire unordered list within the “nav” div) element as your selector and add the following CSS rules:
   * background-color: grey;
     + Note: Whenever you float elements, it collapses their “css box” – thus, if you want to change the background color/image, you must specify a width and height.
     + In this particular case, changing these properties to “width:auto” and “height:20px” should do the trick – they can always be adjusted later.
   * width: auto;
   * height: 20px;
   * overflow:hidden;
3. Make your nav bar more visually appealing by adjusting the padding between your links, and changing the font color. In your CSS, use the **“.nav ul li a”** (individual list item > link) element as your selector and add the following CSS rules:
   * padding: 10px;
   * color: white

**CENTERING YOUR NAV BAR:**

**In order to center your nav bar across the page: since you used the “float:left” property, there are only two ways to “center” your nav bar:**

1. Increase the padding between your links/list items
2. Set the width of your links/list items, then use “text-align:center”
   1. Ex. If your page is 1892px wide (which, by default, it is), and you have 5 links in your nav bar, divide 1892 by 5 (don’t forget to minus ~10px per link for default padding (or set your padding to 0px)
3. Add some pseudo-classes to create more user interactivity and special effects [if this is your first time reading this information, see the “Styling Link States,” “Pseudo-classes,” and “pseudo-elements” sections further down in this document before completing this step].In your CSS add and style pseudo-classes (they don’t need to be the same as the ones listed below) that will alter the “visited,” “hover,” and “active” states of the link, **in that order**:
   * .nav ul li a:visited {color:black;}
   * .nav ul li a:hover {background-color:darkcyan;}
   * .nav ul li a:active {font-weight:bold;}

T Tip/Trick: FYI for some properties, it makes no difference whether they are applied to the “div.nav” “ul” “li” “a” or “li a.” For others, it does. If you are unsure, just move the code around until you’re satisfied with the results!

Tip/Trick: To make your navigation buttons more defined, you can add borders around the entire “ul” element and/or individual “li” items.

Tip/Trick: To center a horizontal Nav Bar, just give the list centered text:

* Example: ul.nav {text-align:center;}

Extension: [Sticky Nav Bar](http://www.w3schools.com/css/css_navbar.asp): Click the link to visit W3School’s tutorial

Extension: [Drop-Down Menus:](http://www.w3schools.com/css/css_dropdowns.asp) Click the link to visit W3School’s tutorial

***Styling Link States -*** *Experiment with W3School’s* [*Styling Navigation Bars Tutorial*](https://www.w3schools.com/css/css_link.asp)

*Link Styles* – Links can be styled just like any other element in your html document using CSS

*Link States* – Links are unique in that, since they are clickable buttons, they have 4 different states of being: **link, visited, hover, active.** These states are determined by YET ANOTHER type of selector (in addition to element, class, id, and descendent) called a **pseudoselector** (which you will learn more about in a few minutes, when you scroll downwards along the page). Here is some more information about each of the 4 states:

* Link (a:link): when the link has not been interacted with (i.e. it has been unvisited)
  + Default = blue, underlined, invisible background
* Visited (a:visited): when the user already visited the link
  + Default = purple, underlined, invisible background
* Hover (a:hover): when the user mouses over the link
  + Default = no difference from link/visited
* Active (a:active): when the link is being clicked on
  + Default = no difference from link/visited

*Link State Order* – As with anything you do when working with HTML and CSS, **order matters**. When setting styles for several link states, it is important that you follow this order (otherwise the styling will not be applied correctly). There is no rhyme or reason to this order – you simply must remember it:

1. a:link
2. a:visited
3. a:hover
4. a:active

*Link Styles* – There are several commonly applied property:value combinations specific to links:

* text-decoration – typically used to remove the underline from links
  + Example: textdecoration:none
  + Other values: overline, line-through, underline (the default)
* background-color – used to specify the background color for a link
  + Example: background-color:yellow
  + Other values: any named, rgb, or hexadecimal color you desire
* display – typically used to define a link as either a block or inline-block to make it button-like
  + Example: display:inline-block (displays each link as its own separate box/item, but positions each link one after the other on the same line, rather than on separate lines)

***Pseudo-Classes -*** *Experiment with W3School’s* [*Pseudo-Classes Tutorial*](https://www.w3schools.com/css/css_pseudo_classes.asp)

*Pseudo-Class*– A special state used to define an element. The state is only activated when interacted with by the user! It could be used in the following scenarios:

* Style an element when a user mouses over it
* Style visited and unvisited links differently
* Style an element when it should receive extra attention

*Formatting Pseudo-Class Rules* – Selector contains BOTH a regular selector AND a pseudoclass.

* Example: selector:pseudo-class { property:value; }
* Example: a:hover { color: red; }

*Types of Pseudo-Classes* – There are many different types of pseudo-classes that are used to control the display of certain CSS elements. However, the only ones we will be learning about are the ones that control links, since they are the most commonly used!

***Pseudo-Elements -*** *Experiment with W3School’s* [*Pseudo-Elements Tutorial*](https://www.w3schools.com/css/css_pseudo_elements.asp)

*Pseudo-Element* – Used in place of the <span> tags to style pre-determined parts of an element. It could be used in the following scenarios:

* Style the first letter, or line, of an element
* Insert content before, or after the content of an element

*Formatting Pseudo-Element Rules* – Selector is followed by 2 colons (::) then the pseudo-element;

* Example: selector::pseudo-element { property:value; }
* Example: p::first-line { color:red; }

**Types of Pseudo-Elements** – There are several different types of pseudo-elements, each of which has its own set of property:value combinations

* selector::first-letter: pseudo-selects the first letter of each specified element
  + Example: “p::first-letter { color:red; }” would make the first letter of each paragraph red
* selector::first-line: pseudo-selects the first line of each specified element
  + Example: “p::first-line { color:red; }” would make the first line of each paragraph red

**The Doing:**

***Navigation Bar Assignment***

Activity Instructions: Copy the “css\_navbars” folder from the Shared > Hammond >Handout and paste it into your CSS folder. Follow the assignment instructions to add a nav bar to the webpage you build in CSS Reviews 1 and 2. You may use either the website you created, or the one included within the folder.

***Comprehension Question:***

* What is a “Nav Bar”?
* What is the difference between a vertical and horizontal nav bar?
* What html element is the foundation of a nav bar?
* What is a “link state”?
* What are the 4 types of link states?
* How is a link state formatted/what does it look like?
* What is a pseudo-class? How is it different from a regular class?
* What is a pseudo-element? What is the advantage to using a pseudo-element?