Advanced CSS Techniques
CA 155 – Personal Website Project

In this lab exercise, we will be working on a website that already has some basic CSS rules applied. We'll be modifying the existing rules and adding some new ones to get a feel for some more advanced CSS techniques that you may end up using in your own projects.

Part Zero: Download the Example Files

To get started with this exercise, download a copy of the example files from: `https://go.wisc.edu/wx2452`

1. Click the Download button in the top right of the page. This will save a .zip file to your computer
2. Open this .zip file to get a copy of the website directory. (You may need to look in your Downloads folder)
   If you're on a Mac, you can just double click the .zip file
   If you're using Windows, you should right click (or Control + Click on a laptop) the .zip file and select "Extract Here…"
3. (optional) Move this folder into your Sites directory to help you stay organized. If you're not planning on using this example after today, it's okay to just delete it once you're finished.

(Be careful about renaming the files – if you change the name of "index.html" or "style.css" the demo website may stop working if you don't also update the `<link>` elements, or similar.

Part One: Gradients for the Background

The `<body>`, `<header>`, and `<main>` elements all have a background-color set. Let's try using a gradient for the backgrounds instead. Let's start with a background gradient for the `<body>`

1. Locate the CSS block that is selecting the body element. Then find the declaration within that block that is setting the background color.
2. Add a new blank line after this declaration. This is where we'll add our declaration to set a gradient.
3. We use the `background-image` characteristic (it's a bit counterintuitive). For the value, we'll be using `linear-gradient()` and specifying the start and end colors in hexadecimal notation. For example, this declaration will create a background gradient that goes from light blue to white:

   ```
   background-image: linear-gradient(#c5f0ef, #FFFFFF);
   ```

   Be sure to keep the `background-color` declaration *above* your gradient declaration. Because in CSS the last rule applied wins, the gradient will display instead of the solid color. But if a user's browser doesn't support gradients, your solid color will display (instead of a boring white background).

   ```
   body{
          background-color: #c5f0ef;
          background-image: linear-gradient(#c5f0ef, #FFFFFF);
   }
   ```

4. Repeat steps 1-3 for the `<header>` and `<main>` elements. Optionally, you can create a new CSS block that selects the `<nav>` elements and sets a different background color or gradient.

(There's more advanced CSS methods on the other side of this sheet)

Part Two: Link Hover Effects

Currently, our website has some basic interactivity. Whenever the user hovers over the items in the <nav>, the text changes to be displayed in italics. Let's modify the rules we already have to make the hovered links stand out even more.

1. Locate the CSS blocks that are selecting the <a> elements within the <nav>
   Hint: The website uses descendent selectors to target *only* certain <a> elements, and not *all* hyperlinks on the page. There are also multiple blocks to select <a> elements that are in different states. Look for things like `a:link`, `a:hover`, and `a:visited` to find the correct CSS blocks
2. Suppose we want to have the navigation links change color whenever the user hovers over a menu item. Add a new declaration to the appropriate CSS block to set a different color value, such as:

   ```
   color: #FF0000;
   ```

3. Suppose we want to have the background color of the navigation link briefly change *while* the user is clicking it. Start by adding a new CSS block to select the <a> elements within the <nav> that are in the "active" state:

   ```
   nav a:active {

   }
   ```

4. Add a declaration to this new code block to set the value of the background-color to something that will stand out. For example:

   ```
   nav a:active {
           background-color: #00ff08;
   }
   ```

Part Three: Basic Animations

Modifiers such as `:hover` and `:active` are known as CSS pseudo-classes, because they modify existing selectors. These pseudo-classes are often used on <a> elements, but can be applied to other elements as well. Let's use `:hover` to add some basic interactivity to the images on the Gallery page.

1. Open "gallery.html" in your code editor if it's not already open. Look at the HTML code to understand how the page content is structured. Note that the <main> element has an attribute that sets `class="gallery"` – we can use this gallery class to select only the images in the gallery, and not other images on our website.
2. In "style.css" locate the CSS block that has a descendent selector to target gallery images only
   Hint: CSS class selectors begin with a period, so this look like `.gallery img { … }`
3. Add a new declaration to this block to control how the browser should animate any changes to this element:

   ```
   .gallery img{
           float: none;
           max-width: 75%;
           transition: transform .2s;
   }
   ```

   The line we just added (italicized above) means that any transformations will be automatically animated to take place over .2 seconds. You can adjust the time to be faster or slower to your liking.
4. Create a new CSS block to select all gallery images that the user is hovering over.
   Hint: The selector should look similar to the block in step 3, but include a `:hover` pseudo-selector. Look ahead to step 5 if you're stuck on how to create this block.
5. Add a transform property to this new block, and set it to zoom the image to 150% of its original size:

   ```
   .gallery img:hover {
           transform: scale(1.5);
   }
   ```