INFO/CS 1300: Lab 3 (9/8)

By Inyoung Hong and Weihong Rong

Due at the end of your lab section. Demo your work in front of a TA to get your participation credit.

Overview

In this lab, we'll go over the basics of CSS as an introduction to styling your webpage. CSS stands for Cascading Style Sheets and controls the layout and appearance of your webpage. Usually, your styling exists as a separate file named **all.css** (or style.css) in a folder named **styles** (or css). While some of you may be familiar with in-line or internal CSS styling, having separate external CSS files make this process much more intuitive and organized.

As discussed in lecture, CSS consists of a set of **properties** and **values** assigned to HTML elements through **selectors**, these properties modify the appearance of the HTML elements so they can take on new colors, shapes, and styles.

What You Need

Bring your **laptop** to your lab section. Please also bring **paper and colored markers, pens, or pencils**. You may also want to **print** out this lab.

Part 1: External CSS

Create a folder called **lab3-<NetID>** (example: lab3-wr85).

Download the lab3.zip file from CMS and unzip the contents to your folder. We have created an HTML file, **index.html** with some exercises to introduce you to some CSS styling tools.

Linking the CSS file

We will need to link our CSS file, **all.css**, to our **index.html**. Linking the two will make any changes you make in the CSS file appear on the webpage. This is pretty similar to linking an image back in Milestone 1.

Open **index.html** with Atom and add the following **link** tag, inside of your **head** tags. Then set the **href** attribute to reference **all.css**.

```
<link rel="stylesheet" type="text/css" href="" media="all"/>
```

This is known as **external CSS** because the styling is in a separate file (i.e. not index.html).

Do not change anything else in index.html for the rest of this lab! This lab is about CSS, not HTML. Open index.html with your browser (or Atom's HTML Preview), the page should now have a light blue background.

Ask for help if you need it; we love helping you!

Part 2: Intro to Styling

Now that you've referenced all.css in index.html, we are ready to start styling this webpage!

Open **all.css** in Atom.

Selectors

Selectors are the *addresses* CSS uses to identify the elements of the HTML files you want to modify. As discussed in class, the most common selectors are **`#id`** and **`.class`**. These selectors, select a particular id or class as specified in the *attributes* of your HTML tags.

Selectors are additive. For example, take the selector `#giraffe-info a`.
`#giraffe-info` selects the HTML element (or tag) with the attribute `id="giraffe-info"`.
`a` selects all of the a elements (or tags).
`#giraffe-info a` selects all of the a elements *inside of* the element with the id of giraffe-info.

Once you have specified your selector, use curly braces `{ }` to specify the styling. Styling is specified with `property: value;` pairs. For example:

```
p {
  color: yellow;
}
```

This will set the color of the text in all your paragraph elements (or tags) to yellow. The curly braces `{ }`, colon `:`, and semi-colon `;` are **required**. So don't forget them!

Task 1: Using ID Selectors & Changing Color

Color in CSS can be specified by color `names`, `rgb(r, g, b)` values, and hexadecimal (`#RRGGBB`) values. We often use hex values because they are convenient and flexible.

Change the color of the `.giraffe-info` selector to **purple**. Then change all of the hyperlinks in the paragraph below to have a font color of **#ff00b5**.

Make your changes to **all.css** under the appropriate comment. **Hint: the selector for all hyperlinks in a document is `a`.**

Save the changes you made to all.css and refresh index.html, you should see that the text is purple and the links have turned pink!

Task 2: Text Styling

Now let's add some color and style to the text in index.html. Text color and styling in CSS is very similar to what you would do in a Word document, but instead of using the graphical buttons in Word, you use CSS to assign values to the various properties.

Scroll down to the section marked “Task 2” in index.html. You should add styling to the paragraphs so that the statements in them are true. For example, the first paragraph should be colored green, have bold text, etc.

We have wrapped the three sentences in three IDs: #green-text, #red-text, and #blue-text. **Use these as your selectors for the task.**

The properties you will use for this task are **color, font-size, font-style, font-weight, text-align, and text-decoration.** To better understand these properties, let’s look at the documentation for *text-align* at https://www.w3schools.com/cssref/pr_text_text-align.asp.

At the top, you will see an example of some possible values, with a “Try it out” button that lets you see how this property can change HTML. Below that is information about the tag. The most relevant for you will be the “Property Values” section, which contains the possible values for this property (*left, right, center, justify, initial*, and *inherit*) and descriptions of what those property values do. You should look at the documentation for all of the properties mentioned above to help you with the task (Hint: underlining text uses the attribute “text-decoration”, while a text-weight of **bold** equals to the tags in HTML).

Task 3: Height and Width

Now let’s move on to the layout of the elements. Widths and height in CSS belongs to two categories, **static** and **relative**. Static dimensions refers to properties assigned to fixed values, while relative ones have proportional values. As their name implies, static dimensions always remain the same size regardless of the size of your window/viewport, while relative ones change proportionally. It is important to consider your use case and choose the appropriate category.

Fixed values are usually implemented in **pixels (px)**, while proportional values are in **percent (%)**.

Now scroll down to the section labeled “Task 3” in index.html. Resize the giraffe head images using static and responsive widths. The top two giraffe images should be selected using the .rel-dim class; you’ll have to look at the HTML in index.html to figure out how to select the bottom giraffe images. When you are done, save your changes to all.css and refresh index.html; **resize the window and see what happens to the two boxes!** (Hint: the static images should remain static and the responsive images should respond to your resizing).

Task 4: Validation

Just like we validate our HTML, we need to validate our CSS. With your all.css file selected in Atom. Open the command palette and search for **w3c validate**. You should see a panel at the bottom of Atom’s window.

Notice that CSS validation often generates lots of warnings. Sometimes we need to address these warnings, but other times the warnings may not be valid for our situation. For your milestone and final submissions you need to validate your HTML and CSS code. **Your validated code should have no errors (0)**; warnings are ok.

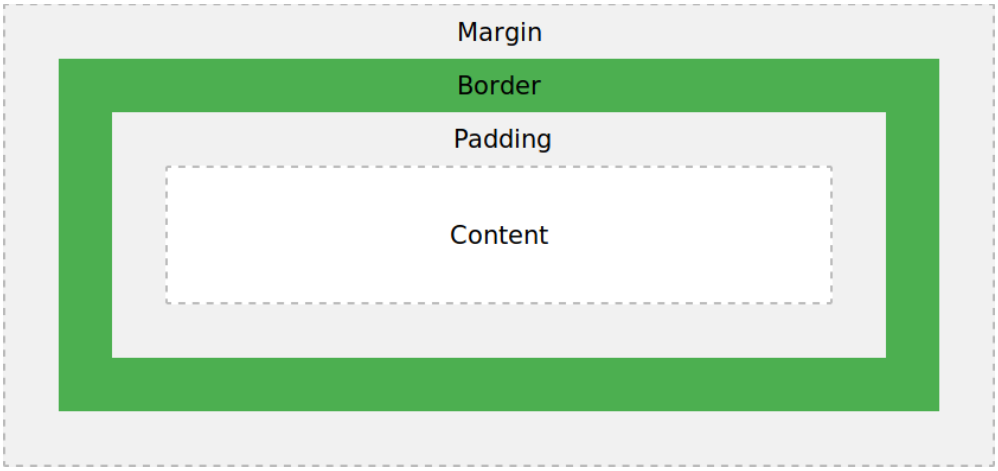
Task 5: Border, Padding, and Margin (Optional)

If you finish Tasks 1-4 before your class has moved on to Part 3, you can start working on Task 5. However, if you don't get through Task 5, that's okay because it's optional!

All elements in CSS follows the **CSS Box Model** as illustrated from the graphic above from w3schools.

Padding and Margins are **transparent** and specify the spacing around content**,** while the borders usually have color and surround the content and padding.

Read about the property and values of the CSS Box Model in the link below and see if you can replicate the image in index.html. Use the skills you just learned in the tasks above, and don't worry about being pixel perfect. https://www.w3schools.com/css/css_boxmodel.asp



Source: https://www.w3schools.com/css/css_boxmodel.asp

Part 3: Designing a Webpage

It's now time to start styling your Project 1. For Milestone 3 you will turn in two separate CSS files with distinctly different styling (not just colors!). While it may be tempting to just start designing your webpages by just writing the CSS, this is a mistake! CSS is extremely complex and often times it renders incorrectly in different browsers. A much smarter approach is to design on paper first, then try to get the CSS to render like your paper design.

Color & Emotion

However, before you can start designing your page, we should discuss color. The choices you make in terms of color communicate *something* to your audience. Color invokes an emotional response in all of us. For example, think about how you feel when you see red ink written all over your homework. How does the red make you

feel? Would you feel the same if the ink was blue or green? By using human's emotional response to color we can communicate more effectively to your website's audience.

As a class, discuss how the following colors make you feel:

- red
- green
- blue
- purple
- pink
- yellow
- orange

Now discuss the same colors, but think of them as the main color for a website. Does blue make you feel confidence in the website? Does red give you a feeling of importance? Does green make you think of the environment or prosperity? Does purple invoke feeling of glamour? Does pink give a perception of imagination? Is yellow fun and happy? Does orange appear playful?

Now think about how light and dark affect your perception. Brighter colors invoke more energy, while darker colors are more relaxing. More saturated colors pop and draw attention from the eye, while less saturated colors fade into the background.

Write down **several adjectives** that you want your audience to perceive when they visit your Project 1 site. Now **write colors next to each adjective**. Discuss with your color choices with your neighbors and see if they have similar responses to the emotions you are trying to invoke for your website.

You may use these exercises as part of your rationale for Milestone 3.

Sketching First, CSS Second

Now that you've thought about some of the emotions and colors for your website. Let's start thinking about the specific design of your site. Remember designing in CSS is usually a bad idea. Often times, you will invest so much time in making your CSS work, you will be less willing to throw the design away if it's bad. It's much better to make some quick sketches on paper first. That way if you sketch out a design that's bad, you've not spent that much time on it, so you don't feel bad about pitching it. Only when you have a sketch that you like, should you start writing the CSS for that design.

Take out several pieces of paper and start sketching several designs. Think about the spacing between the structural elements on your webpage. Think about color and either use colored pens/markers or write the color names on the sketch. Think about font sizes and font families. Sketch all of this out in wire-frames and annotate the design as necessary. **Sketch out a least 3 separate designs**.

You should definitely use these sketches in your Milestone 3 rationale. We want to see how you thought about your audience and how your design reaches them.

Credit for Lab 3

Demo your work for a TA. You should demo your CSS styling and correct any mistakes the TA points out. You should also show the TA your sketches for Project 1, Milestone 3.

Additional Resources

This lab activity only covers the very basics of CSS, but we want this activity to serve as a start to your explorations into styling your website. Remember that CSS is a powerful set of tools that help you achieve your vision for your site, but the best way to design a page is drawing out a wire frame. Remember, **don't use CSS to design your page, use it to make your design possible.** Below are some resources so you can learn more about the numerous attributes of CSS and how to apply them for your site.

W3Schools CSS Resources

An integral part of CSS is understanding the various properties that allows you to realize your wire-framed designs. As we cannot exhaustively go through the list of all the properties of CSS since the list is so massive, it becomes a great skill in web design to be able to go through them yourselves. Luckily, they all (generally) follow the same CSS format as the properties we have gone through today in the lab activity.

In the link below you can find a dictionary of all properties in the current implementation of CSS. Experiment with these properties and have fun! However, do note the varying degrees of browser support for some of the more obscure properties and exercise restraint in their use. https://www.w3schools.com/cssref/default.asp

CSS Diner

We know selectors can be tricky to understand, especially if this is your first time dealing with them. CSS Diner is a game(!) that makes picking up the nuances of selectors interactive and pretty fun. If you finish the lab early, feel free to help out your neighbors or have some fun playing this interactive minigame:
https://flukeout.github.io

Trust us in that selectors will become easier as you use them more and more and grab the logic behind them. If you prefer a more textual and detailed account on them, w3schools offers a great reference page:
https://www.w3schools.com/cssref/css_selectors.asp