Notes for the professor:

How can I put my website that I’ve created for the hw to school’s github page?

**Html (insert the link)**

<head>

<title>Vacation World</title>

<link href="style.css" type="text/css" rel="stylesheet">

</head>

and

href="http://travel.nationalgeographic.com/travel/city-guides/seoul-south-korea/" target="\_blank">Learn More</a>.

Html (insert an image)

<img src="https://s3.amazonaws.com/codecademy-content/courses/freelance-1/unit-2/explorer.jpeg" />

**STYLIZE: HTML+CSS**

Html

<h1 class="cursive capitalize" >Top Vacation Spots</h1>

css

.cursive {

font-family:cursive;

}

.capitalize {

text-transform:capitalize;

}

**Html bullet points:**

<h5>Top Attractions</h5>

<ul>

<li>Museums</li>

<li>Bike Tours</li>

<li>Historical Monuments</li>

</ul>

</div>

**Chaining and Specificity**

h1.special { }

The code above would select only the h1elements that have a class of special. If a p element also had a class of special, the rule in the example would not style the paragraph.

Or

.description h5 {

color: teal;

}

In the last exercise, instead of selecting all h5elements, you selected only the h5 elements nested inside the .description elements. This CSS selector was more specific than writing only h5. Adding more than one tag, class, or ID to a CSS selector increases the specificity of the CSS selector.

For instance, consider the following CSS:

p { color: blue; }

.main p { color: red; }

**!important**

Notice that the h5 elements will now be rebeccapurple instead of teal. That's because !important will override any other style no matter what.

!important will hold you color/style etc. no matter what

# Multiple Selectors

h1 { font-family: Georgia; }

.menu { font-family: Georgia; }

Instead of writing font-family: Georgiatwice for two selectors, we can separate the selectors by a comma to apply the same style to both, like this:

h1,

.menu { font-family: Georgia; }

By separating the CSS selectors with a comma, both the h1 and the .menu elements will receive the font-family: Georgia styling.

**Conclusion:**

Throughout this lesson, you learned how to select HTML elements with CSS and apply styles to them. Let's review what you learned:

* CSS can change the look of HTML elements. In order to do this, CSS must select HTML elements, then apply styles to them.
* CSS can select HTML elements by tag, class, or ID.
* Multiple CSS classes can be applied to one HTML element.
* Classes can be reusable, while IDs can only be used once.
* IDs are more specific than classes, and classes are more specific than tags. That means IDs will override any styles from a class, and classes will override any styles from a tag selector.
* Multiple selectors can be chained together to select an element. This raises the specificity, but can be necessary.
* Nested elements can be selected by separating selectors with a space.
* The !important flag will override any style, however it should almost never be used, as it is extremely difficult to override.
* Multiple unrelated selectors can receive the same styles by separating the selector names with commas.

# Lesson #2: CSS Structure;

* The property and value are separated by a colon (:). A semicolon (;) should always be used at the end of a declaration.
* When the name of a typeface consists of more than one word, it's a best practice to enclose the typeface's name in quotes, like so:

h1 { font-family: "Courier New"; }

# Opacity

Opacity is the measure of how transparent an element is. It's measured from 0 to 1, with 1 representing 100%, or fully visible and opaque, and 0 representing 0%, or fully invisible.

Opacity can be used to make elements fade into others for a nice overlay effect. To adjust the opacity of an element, the syntax looks like this:

.overlay

{ opacity: 0.5; }

In the example above, the .overlay element would be 50% visible, letting whatever is positioned behind it show through.

# Background Image

CSS has the ability to change the background of an element. One option is to make the background of an element an image. This is done through the CSS property background-image. Its syntax looks like this:

.main-banner {

background-image:

url("https://www.example.com/image.jpg"); }

The url should be a url to an image. The url can be a file within your project, or it can be a link to an external site. To link to an image inside an existing project, you must provide a relative file path. If there was an image folder in the project, with an image named mountains.jpg, the relative file path would look like:

.main-banner

{ background-image:

url("images/mountains.jpg"); }

# Conclusion: Review Visual Rules

Incredible work! You used CSS to alter text and images throughout a website. Throughout this lesson, you learned concepts including:

* CSS declarations are structured into property and value pairs.
* The font-family property defines the typeface of an element.
* font-size controls the size of text displayed.
* font-weight defines how thin or thick text is displayed.
* The text-align property places text in the left, right, or center of its parent container.
* Text can have two different color attributes: color and background-color. color defines the color of the text, while background-color defines the color behind the text.
* CSS can make an element transparent with the opacity property.
* CSS can also set the background of an element to an image with the background-image property.

**HTML**

The <!DOCTYPE html> declaration provides the browser with two pieces of information (the type of document and the HTML version to expect), but it doesn't actually add any HTML structure or content.

To create HTML structure and content, we must add opening and closing <html> tags after declaring <!DOCTYPE html>:

<!DOCTYPE html>

<html>

</html>

Anything between the opening <html> and closing </html> tags will be interpreted as HTML code.

* <!DOCTYPE html>, the declaration specifying the version of HTML for the browser
* The <html> tags that enclose all of your HTML code
* The <head> tag that contains the metadata of a webpage, such as its <title>

**Hyperlink:**

<a href="https://www.wikipedia.org/">This Is A Link To Wikipedia</a>

<a

href="https://en.wikipedia.org/wiki/Brown\_bear">Learn More</a>

# Opening Links in a New Window:

For a link to open in a new window, the target attribute requires a value of \_blank. The target attribute can be added directly to the opening tag of the anchor element, just like the href attribute.

<a href="https://en.wikipedia.org/wiki/Brown\_bear"

target="\_blank">The Brown Bear</a>