"CSS Core Concepts" **CCC 1**

***CSS Basics***

1) What risks do we run as developers if we ignore the default styles of the browser?

We run the risk of unwanted formatting errors that cause our page to display differently than intended and may be hard to track down.

2) [Critical Thinking] What is a CSS reset? Do you use one?  Why or why not?

The video states: “A CSS reset is a collection of styles that zero out the formatting from these browser default style sheets, and they help to give you a consistent point from which to begin writing your own styles.”. I personally use them sparingly and for specific elements that I want to target and style. I do this because sometimes I want an element to display a certain way and rather than risk a potential mess due to not knowing what browser is going to be used, if I reset I can ensure that it will be displayed the way I intended it to be.

3) In CSS what is the purpose of the declaration?

They give the formatting instructions for the style.

4) The author mentions several reasons to use comments in CSS.  Name two.

a) organizing styles

b) annotating code

**Targeting Page Content**

5) [Critical Thinking] When the author talks about focusing on **semantics** when selecting tags, what does he mean?

He means that you should use the tag that best expresses the type of content contained within it in a meaningful way. This allows content to be styled easier and more consistently.

6) [Critical Thinking] How does the author feel about styling elements wrapped in multiple **div** tags?

The author feels that this is unnecessary and extraneous markup and that the same visual effect could be achieved without the added structure.

7) [Critical Thinking] What is the author’s opinion about web designers using classes and IDs as simply a means of styling page content?

His opinion is that this is a mistake because this is not the primary purpose of these attributes. The author states: “Class and ID attributes exists to help give your content more meaning.”

8) What is the connection between **IDs** and **JavaScript**?

The connection is that “JavaScript allows us to target elements on the page that have an ID and then do something like style it, change the content, or modify it in some way, or attach some type of behavior to it.”

9) [Critical Thinking] Why does the author identify the main regions on the page using IDs?

In order to semantically identify and easily target with styles, also to add additional meaning to elements.

10) [Critical Thinking] What is the potential negative effect of the universal selector on **inheritance**?

The potential negative effect of the universal selector on inheritance is that it is applied to every single element on a page like buttons, and input, text fields, and checkboxes, and overrides even styles applied to their child elements.

11) [Critical Thinking] The author says he never needs to use the universal selector.  What does he do instead?

What the author does instead of using the universal selector is that he uses a descendant selector instead.

12) [Critical Thinking] When using descendant selectors does **white space** matter?  Why or why not?

Yes, using a space between the elements listed in this example: aside p{} tells the browser to apply the style to any paragraph within any aside element regardless of whether the paragraph is inside of other elements within the aside itself.

13) [Critical Thinking] Write an example CSS rule that uses a descendant selector to target all **p** tags inside an **article** element with an id of **sports**.

article#sports p{}

14) [Critical Thinking] What is the difference between an **adjacent** selector and a **descendant** selector?

An adjacent selector lets you style in an element based on which element comes before it in the document, providing that both of those elements are inside the same parent element. A descendant selector styles descendants of the parent no matter where they are within the parent element.

15) [Critical Thinking] Give an example of something you can do with an adjacent selector that would be hard to do with a descendant selector?

The example in the video was styling ***only a paragraph immediately following*** a heading 2 tag when they are within the same parent element. Using a descendant selector would style ***all*** of the paragraphs after a heading 2 tag.

16) Do pseudo class selectors fall outside of the DOM?

The states: ” Pseudo-class selectors allow us to target elements that lie outside of the DOM…”

17) [Critical Thinking] Write a pseudo class selector that only affects the **hover** on **a** tags inside all section elements with a class of **flyout**.

.flyout a:hover{}

18) What does the mnemonic LVHA stand for?

L- link

V- hover

H- visited

1. active

19) [Critical Thinking] Write a CSS rule that turns the **background** of only the **odd rows** in all **TR**s in a table with an **ID** of **email**.

Table#email tr:nth-child(odd){ background:red;}

20) What does a **negative number** do with the **nth child** selector?

It starts at the number element you specify and then applies it to the ones before it. (E.g.: li:nth-child(-2n+6){} would apply the styling to the second, fourth and sixth list items.) if you use a negative like so: (5n-1) it will start at the fifth element, then move up to the fourth and apply the style. Then it will apply the style to every fifth element after that, so, 4th, 9th, 14th items and so on.

21) [Critical Thinking] What can you do with a pseudo element selector that you can’t do otherwise?

The video says: “Pseudo-elements are similar to pseudo- classes in many ways, as they allow you to access content beyond the normal capabilities of the DOM. For example, using pseudo-element selectors, you could target the first line or first letter of an element. Pseudo-element selectors also allow us to create what is known as generated content, that is, actually place content on the page that is not contained in the structure of the document.” And it allows us to do this without any additional markup.

22) [Critical Thinking] Write a selector that adds the text **“Note: “** before the content of every **P** element whose class attribute has the value of **“note”**.

p.note::before{ content: ”Note: ”;}