Understanding Order and Specificity in CSS

When you write your styles and something seems not to work as expected, you may want to check your style "order" and "specificity".

Let’s pull up the example from a previous lesson:

Output

HTML

CSS (SCSS)

```scss
body {
  background-color: #ccc;
}

h1 {
    color: black;
}

.info {
    color: white;
}

.primary {
    color: green;
}

#warning {
  color: red;
}

.info,
.primary {
    background-color: black;
}
```

Be sure to check out the code above.

I'll now explain the concepts of order and specificity with an example.

In the code above, what happens if I added a conflicting declaration to the `h1` element?

Right now, we have this:

```
h1 {
  color: black;
}
```

A conflicting declaration has been added in the code output below. Take a look at the highlighted line.

Output

HTML

CSS (SCSS)

```
body {
  background-color: blue;
}

h1 {
    color: black;
  color: white;
}

.info {
    color: white;
}

.primary {
    color: green;
}

#warning {
  color: red;
}

.info,
.primary {
    background-color: black;
}
```

It doesn’t make sense to write CSS like this. However, see `Line 7` above. `Line 6` says, *“give the h1 a color of black”*. On the contrary, the line just after that says, _“give `h1` a color of white”

What does CSS do in this awkward case?

Yeah, `Line 7` wins!

Check the output tab above. You’ll see the `h1` with a color of white!

Why?

This is because in CSS, the `order` in which you write your declarations matter. If a conflicting declaration is found further down the styles document, it is honored.

Now that’s order explained!

Exercise

Sometimes conflicts don’t exist in the same code block `{...}`
What if a conflicting code block existed further down the style document?

Delete `Line 7` in the code above. At the very end of the style sheet, `Line 25` write this:

`h1 { color: orange; }`

What do you think happens? Try it now.

Specificty in CSS

This is the second part of this lesson. I don’t want you to think too hard about specificity for now. As you get more experienced with CSS you’ll know their place a lot better.

Consider the markup below

```
<html>
  <head></head>
  <body>
    <header>
      <h1>I love CSS!</h1>
    </header>
  </body>
</html>
```

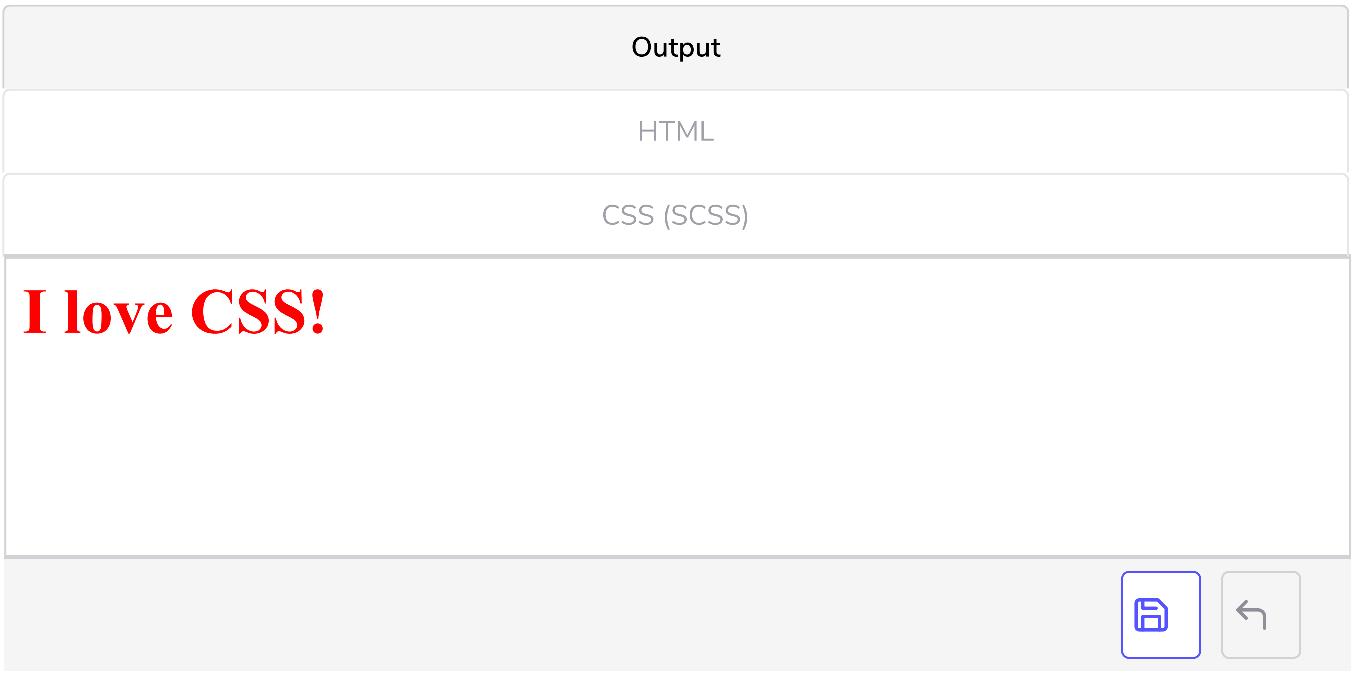
As a refresher, the `header` tag is an `html5` tag. Mostly used as a container for introductory content on a page.

So let's try to style the `h1` element. This time in a way different from any ways I have shared so far.

```
header h1 {
  color: red;
}
```

`header h1` selects every `h1` in a `header` tag. It's like saying, *"select every h1 in a header tag"*

As expected this would give a red color.



I love CSS!

Good.

What if we include a conflicting css rule in this form:

```
h1 {
  color: blue;

}
```

Which of the CSS rules is honored?

Output

HTML

CSS (SCSS)

```
header h1 {
  color: red;
}

h1 {
  color: blue;
}
```

From the last lesson, you would expect the color of `h1` to be `blue`. But, in this case that is wrong.

Click the output in the code above to view the result.

Red it is!

So, why?

Specificity.

Explanation

If you know a thing about sports, you must know that the team with the highest points is declared winner at the end of the season.

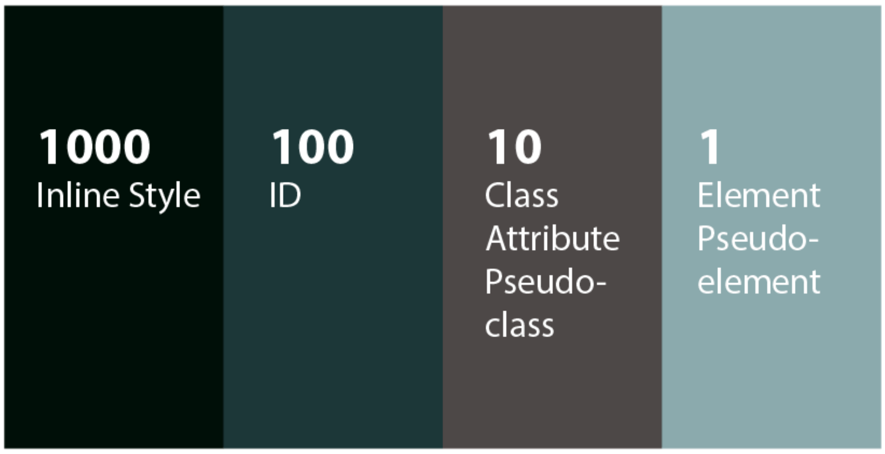
That is (in someway) how specificity in CSS works.

The selector `header h1` has more points than `h1` so its declaration, `color: red` wins over `color: blue`

How are the Points Calculated

The points are calculated off of the selectors in the CSS rule. In the example above, the selectors compared are `header h1` and `h1`

Below is an graphic that shows the points attributed to each kind of selector.



The Calculation Explained

How exactly does `header h1` have a higher point than `h1`? Let’s take a look.

`header h1` contains 2 elements. `header` and `h1`. An element gets 1 point (see the graphic above). Thus, this results in 2 points.

`h1` has just one element. This results in only 1 point.

`header h1` is the obvious winner here, as 2 is greater than 1.

Like I said earlier, don’t think hard about this now. It is great to be aware of the consequences though.

To calculate specificity a lot more visually, this specificity calculator is very handy