Cascading Style Sheets (CSS)

What is CSS?

CSS is a stylesheet language used to describe the presentational semantics of a document written in a markup language like HTML.

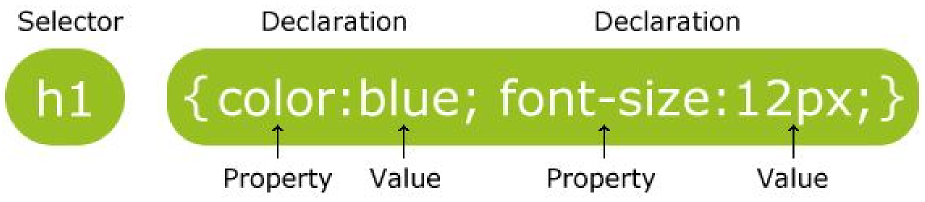
Brief Overview

HTML was never intended to contain tags that would give style to a document. It was always to describe the content of documents. However in HTML 3.2, some presentational tags were added such as `<center>` or `<font>`. While it was nice to have some control on the look of websites, it was a challenge to make uniform changes even in the smallest of websites. Imagine manually selecting every link on the website and individually tweaking the color. To solve this problem, the W3C developed CSS.

In HTML4, all of the formatting could be removed from the HTML document and placed in a separate style document called CSS. External stylesheets enable you to change the appearance and layout of every webpage on a website just by editing one single file.

CSS Syntax:

A CSS rule-set consists of a selector and a declaration block:



- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a CSS property name and a value, separated by a colon.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

“Cascade” in Cascading Style Sheets

Vocabulary:
- Cascade: a process whereby something, typically information or knowledge, is successively passed on

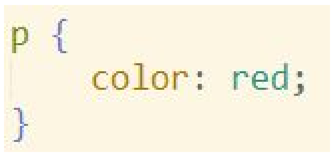
3 main ways styles cascade:

1. The first way stylesheets cascade, is that **a single style can affect multiple HTML elements**.

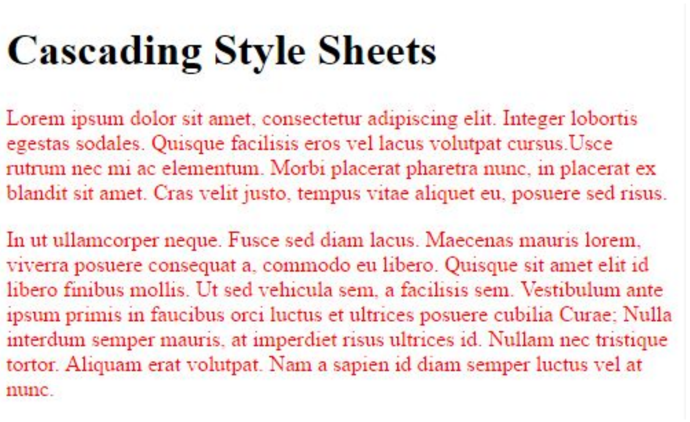
   For example, let's say we have a few paragraph tags. Then let's say that we change our mind and we want all of our paragraphs on our website to be red. We can write one simple style to do this.

   All of the elements that match the criteria of that style rule, will be affected by it. In this case, every paragraph. Notice that our heading was not affected.

   Example:



```
p {
    color: red;
}
```



Cascading Style Sheets

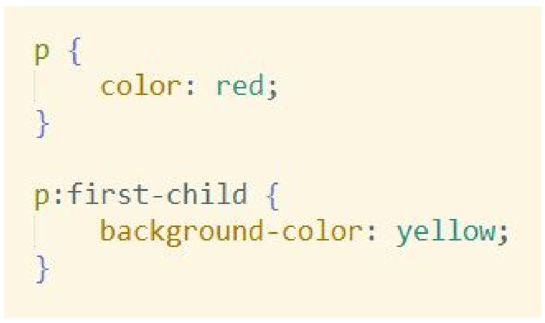
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lobortis egestas sodales. Quisque facilisis eros vel lacus volutpat cursus.Usce rutrum nec mi ac elementum. Morbi placerat pharetra nunc, in placerat ex blandit sit amet. Cras velit justo, tempus vitae aliquet eu, posuere sed risus.

In ut ullamcorper neque. Fusce sed diam lacus. Maecenas mauris lorem, viverra posuere consequat a, commodo eu libero. Quisque sit amet elit id libero finibus mollis. Ut sed vehicula sem, a facilisis sem. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nulla interdum semper mauris, at imperdiet risus ultrices id. Nullam nec tristique tortor. Aliquam erat volutpat. Nam a sapien id diam semper luctus vel at nunc.

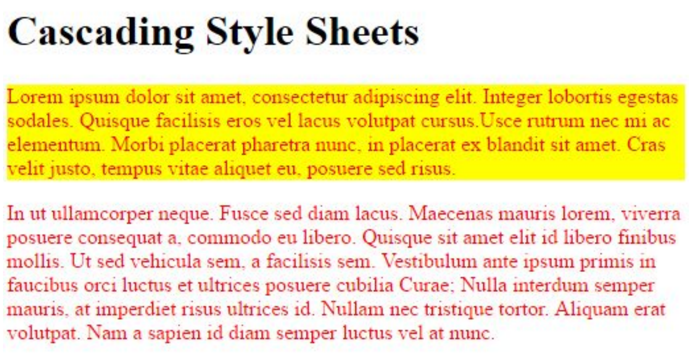
2. The second way that stylesheets cascade is similar to the first, but reversed. That is, **one html element can be affected by many styles**, in different ways, all at the same time.

   Let's say we write a general style that makes all of our paragraphs have red text, and then later on we target the first paragraph and make it 18px large. It is now both red and 18px large.

   Example:



```
p {
    color: red;
}

p:first-child {
    background-color: yellow;
}
```



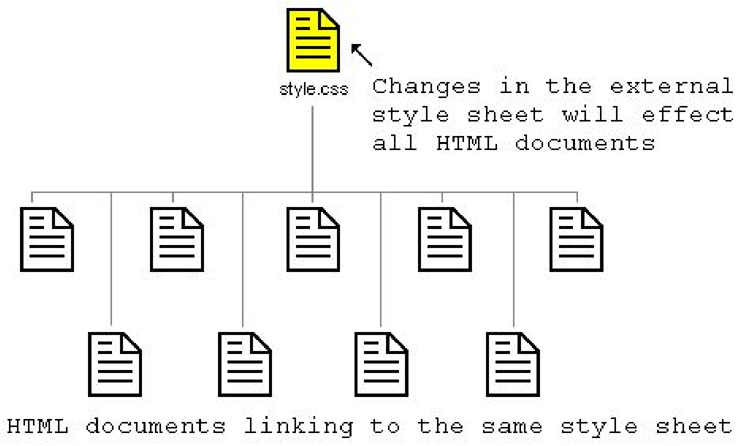
Cascading Style Sheets

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lobortis egestas sodales. Quisque facilisis eros vel lacus volutpat cursus.Usce rutrum nec mi ac elementum. Morbi placerat pharetra nunc, in placerat ex blandit sit amet. Cras velit justo, tempus vitae aliquet eu, posuere sed risus.

In ut ullamcorper neque. Fusce sed diam lacus. Maecenas mauris lorem, viverra posuere consequat a, commodo eu libero. Quisque sit amet elit id libero finibus mollis. Ut sed vehicula sem, a facilisis sem. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nulla interdum semper mauris, at imperdiet risus ultrices id. Nullam nec tristique tortor. Aliquam erat volutpat. Nam a sapien id diam semper luctus vel at nunc.

3. The third way they cascade is what allows us to use the powers of stylesheets to make large scale websites. In this way, **one stylesheet can be applied to many pages across a website**, or even multiple sites!

   For example, on my website I have many pages, on all of those pages I have the same header. This is easy to accomplish with CSS. I just link all the HTML pages to that same style sheet.


style.css
Changes in the external style sheet will effect all HTML documents
HTML documents linking to the same style sheet

Selectors

Selectors are used in CSS to identify specific HTML elements in order to give them styling that they need.

Basic Selectors:

- **Type Selector** - matches all elements of that name on the page

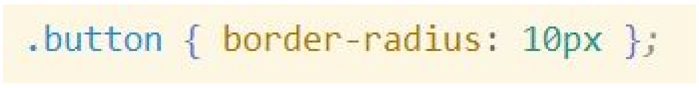
```
h1, p {
    font-size: 10px;
}
```

- **ID Selector** - matches all elements that have an ID attribute with the value specified
  - ID's are unique
    - Each element can have only one ID
    - Each page can have only one element with that ID
    - Select ID's by using a #.
    - Example: (find that one element whose ID's name is button)

```
#button { border-radius: 10px };
```

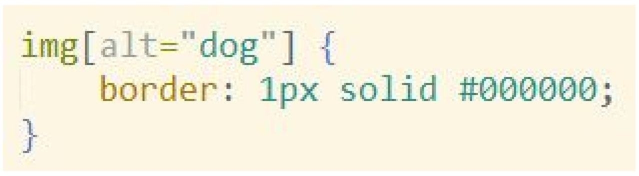
- **Class Selector** - matches all elements that have a class attribute with the value specified
  - Classes are not unique

- You can use the same class on multiple elements.
- You can use multiple classes on the same element.
- Select Classes using a `. (dot)`
- Example: (find all of the elements which have a class name of button)



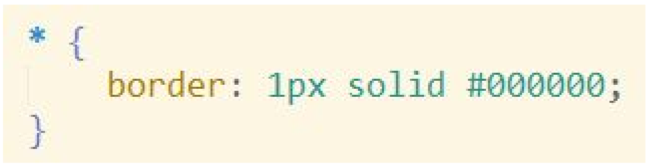
```
.button { border-radius: 10px };
```

- **Attribute selector** - Selects elements based on their attributes



```
img[alt="dog"] {
    border: 1px solid #000000;
}
```

- **Universal selector** - Selects every element on the page.



```
* {
    border: 1px solid #000000;
}
```

Properties and Values

The Semantics look like this:

```
selector {
	property: value;
}
```

3 Ways to Insert CSS

1. **External Style Sheet**
   - With an external style sheet, you can change the look of an entire website by changing just one file
   - The style sheet file must be saved with a .css extension.
   - Syntax:

```
<head>
	<link rel="stylesheet" type="text/css" ref="mystyle.css">
</head>
```

2. **Internal Style Sheet**
   - An internal style sheet may be used if one single page has a unique style.
   - Internal styles are defined within the `<style>` element, inside the `<head>` section of an HTML page:

```
<head>
      <style>
            ...
      </style>
</head>
```

3. **Inline Style**
   - An inline style may be used to apply a unique style for a single element
   - To use inline styles, add the style attribute to the relevant element

```
<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>
```

CSS3 Property Groups

● Color ● Background and Borders ● Basic Box ● Flexible Box ● Text ● Text Decoration ● Fonts ● Writing Modes	● Table ● Lists and Counters ● Animation ● Transform ● Transition ● Basic User Interface ● Multi-column	● Paged Media ● Generated Content ● Filter Effects ● Image/Replaced Content ● Masking ● Speech ● Marquee

Combinators

A combinator is something that explains the relationship between the selectors.

Four different combinators in CSS3:

- Adjacent sibling selector (+)
- General sibling selector (~)
- Child selector (>)
- Descendant selector (space)

1. **Adjacent sibling selectors**

   Also known as the *next-sibling selector*. It will select only the specified element that immediately follows the former specified element. Sibling elements must have the same parent element, and "adjacent" means "immediately following".

   Syntax: `former_element + target_element { style properties }`

Example:

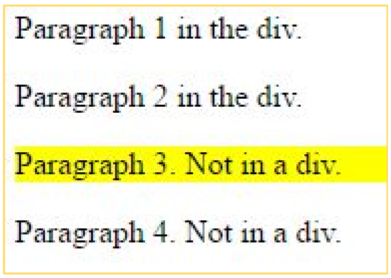
Style:

```
div + p {
    background-color: yellow;
}
```

HTML:

```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
</div>

<p>Paragraph 3. Not in a div.</p>
<p>Paragraph 4. Not in a div.</p>
```

Output:



```
Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3. Not in a div.

Paragraph 4. Not in a div.
```

2. **General sibling selectors**

   The '~' combinator selects elements that follow (not necessarily immediately) the former specified element, if both elements shared the same parent.

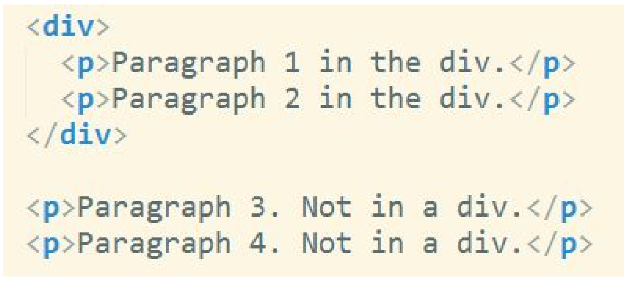
   Syntax: `element ~ element { ` *`style properties`* ` }`

   Example:

Style:

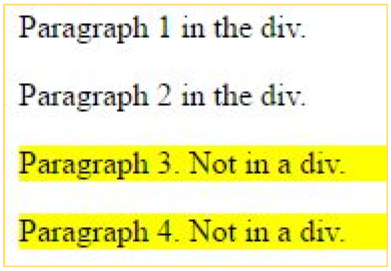
```
div ~ p {
    background-color: yellow;
}
```

HTML:



```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
</div>

<p>Paragraph 3. Not in a div.</p>
<p>Paragraph 4. Not in a div.</p>
```

Output:



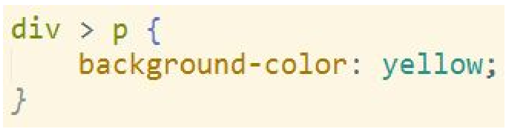
```
Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3. Not in a div.

Paragraph 4. Not in a div.
```

3. **Child selectors**
   The '>' combinator selects elements that are direct children of the former specified element.

   Syntax: `selector1 > selector2 {` *`style properties`* `}`

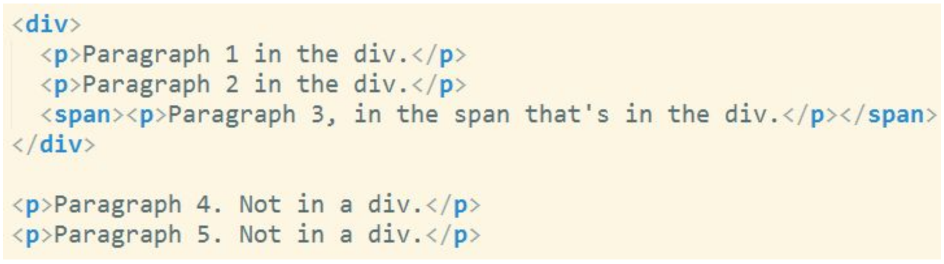
   Example:

Style:



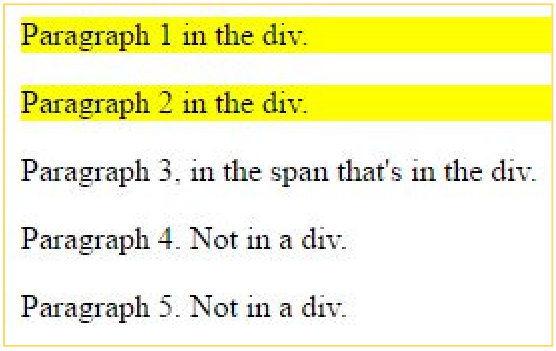
```
div > p {
    background-color: yellow;
}
```

HTML:



```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <span><p>Paragraph 3, in the span that's in the div.</p></span>
</div>

<p>Paragraph 4. Not in a div.</p>
<p>Paragraph 5. Not in a div.</p>
```

Output:



```
Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3, in the span that's in the div.

Paragraph 4. Not in a div.

Paragraph 5. Not in a div.
```

4. **Descendant selectors**
   The ' ' combinator selects nodes that are children (not necessary direct children) of the former specified element.

   Syntax: `selector`$_1$ `selector`$_2$ `{` *`style properties`* `}`

   Example:

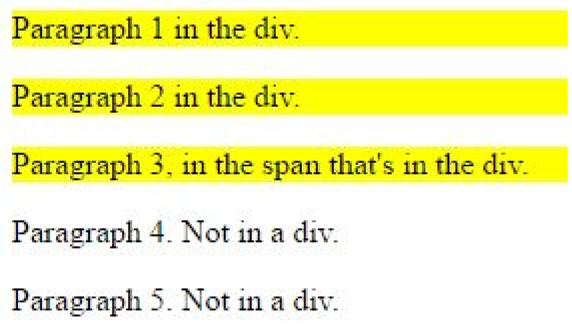
   Style:

```
div p {
    background-color: yellow;
}
```

   HTML:

```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <span><p>Paragraph 3, in the span that's in the div.</p></span>
</div>

<p>Paragraph 4. Not in a div.</p>
<p>Paragraph 5. Not in a div.</p>
```

   Output:



```
Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3, in the span that's in the div.

Paragraph 4. Not in a div.

Paragraph 5. Not in a div.
```

Pseudo-classes
Pseudo-classes are used to provide styles not for elements, but for various states of elements.

- `:link` — the normal, default state of links, just as you first found them.
- `:visited` — selects links that you have already visited in the browser you are currently using.
- `:focus` — selects links that currently have the keyboard cursor within them.
- `:hover` — selects links that are currently being hovered over by the mouse pointer.
- `:active` — selects links that are currently being clicked on.

nth selector

- `:first-child` — represents any element that is the first child element of its parent.
- `:last-child` — represents any element that is the last child element of its parent.
- `:first-of-type` — represents the first sibling of its type in the list of children of its parent element.
- `:last-of-type` — represents the last sibling with the given element name in the list of children of its parent element.

- `:nth-child(n)` — represents every element that is the nth child of its parent

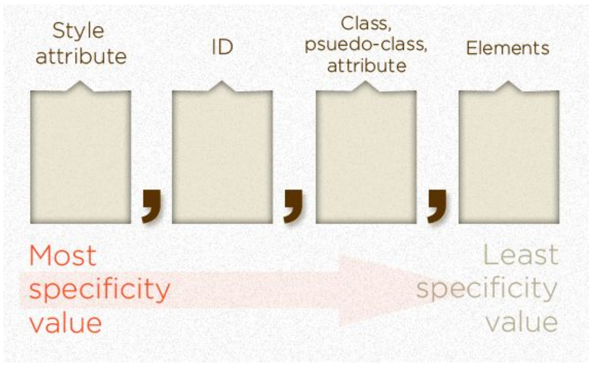
CSS Specificity

Overview:

1. Specificity determines which CSS rule is applied by the browsers.
2. Specificity is usually the reason why your CSS-rules don't apply to some elements, although you think they should.
3. Every selector has its place in the specificity hierarchy.
4. If two selectors apply to the same element, the one with higher specificity wins.
5. There are four distinct categories which define the specificity level of a given selector: inline styles, IDs, classes, attributes, and elements.
6. When selectors have an equal specificity value, the latest rule is the one that counts.
7. When selectors have an unequal specificity value, the more specific rule is the one that counts.
8. The last rule defined overrides any previous, conflicting rules.
9. The embedded style sheet has a greater specificity than other rules.
10. ID selectors have a higher specificity than attribute selectors.
11. A class selector beats any number of element selectors.
12. The universal selector and inherited selectors have a specificity of 0, 0, 0, 0.
13. You can calculate CSS specificity with CSS Specificity Calculator.

How to measure specificity?

- Add 1000 for style attribute
- add 100 for each ID
- add 10 for each attribute, class or pseudo-class
- add 1 for each element name or pseudo-element.



So in `body #content .data img:hover` the specificity value would be 122 (0,1,2,2 or 0122):
100 for `#content`, 10 for `.data`, 10 for `:hover`, 1 for `body` and 1 for `img`.

Alternative way: "Count the number of ID attributes in the selector (= a). Count the number of other attributes and pseudo-classes in the selector (= b). Count the number of element names and pseudo-elements in the selector (= c). Concatenating the three numbers a-b-c gives the specificity.