CSS Syntax

You've seen many examples of CSS syntax in this lesson. Here's a review!

The basic syntax of a CSS ruleset has two parts: a *selector*, and a group of *rules*, each of which consists of a *property* name and the *value* of that property.

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

The selector is written first, and then the rules are written inside `{` curly brackets `}`. Each rule's property and value are separated by a `:` colon, and the rule always ends with a `;` semicolon.

Selectors

The selector indicates which HTML elements the rule will apply to. You've seen a few different sorts of selector: the *element* selector, the *class* selector, the *id* selector, and the *descendant* selector.

A **type selector** applies to every HTML element of a particular type, such as `p` or `em`. This selector will apply to every `p` element:

```
p {
    color: blue;
}
```

A **class selector** applies to all elements that share a `class` attribute. The class selector is written starting with a `.` (dot):

```
.narrow {
    width: 20%;
}
```

In order for the class selector to apply, there have to be HTML elements on the page that use that `class` attribute:

```
<div class="narrow">
   This will get the 20% width.
</div>
```

An **id selector** applies to an element with a particular `id` attribute. The id selector is written starting with a `#` sign:

```
#sidebar {
    background-color: lightgray;
    width: 20%;
    float: left;
}
```

Within an HTML page, there should be only one element with that `id` attribute value.

```
<div id="sidebar">
    This will get the background, width, and float values from the sidebar CSS rule.
</div>
```

A **descendant selector** is a compound of two simpler selectors. It applies only to an inner element that is a descendant (on the DOM tree) of a particular outer element.

```
li a {
    color: pink;
}
```

The above selector will apply to `a` elements (hyperlinks), but only those inside an `li` element (list item):

```
<ul>
    <li> <a href="https://www.udacity.com/"> Pink Udacity </a>
</ul>
<p> <a href="https://www.google.com/"> Non-pink Google </a>
```

Rules

A ruleset can be composed of several rules, each of which applies a particular *value* to a *property* of the selected elements. Properties are things such as the color, position, size, and shape of the element.

```
h1 { color: red; font-size: larger; }
```

This rule applies the value `red` to the property `color`, and the value `larger` to the property `font-size`.

Some properties allow values that are more than one word long, such as the `font` property:

```
body { font: 12pt bold Consolas, Monaco, monospace; }
```

Font Stacks

The `font-family` and `font` properties allow you to specify a *font stack*, a list of font options separated by `,` commas. The browser will use the first font in the stack that is available on the user's system. Usually the last font in the stack should be a generic font name, such as `serif`, `sans-serif`, or `monospace`.

Colors

There are several ways to specify a color in CSS. Three common ones are hex codes, `rgb` triples, and color names.

```
.orange {
    color: #ff9900;
}
.pink {
    color: rgb(100%, 80%, 80%);
}
.chartreuse {
    color: chartreuse;
}
```

Flexbox

To change the browser's layout from the default document-based layout to the flexible box layout, set `display: flex` on a container element (one that has other elements inside it).

```
.outer {
    display: flex;
    border: 2px dotted orange;
}
.inner {
    width: 100px;
    border: 1px solid black;
    padding: 10px;
}
```

Flexbox can be heavily customized! The above will cause `.inner` HTML elements to be packed in a row within the `.outer` element:

```
<div class="outer">
   <p class="inner"> I am a box. </p>
   <p class="inner"> I am another box. </p>
   <p class="inner"> Hey, I am a box, too! Boxes
<strong>rock</strong>. </p>
   <p class="inner"> Let's be boxes together. Yay, flexbox. </p>
</div>
```