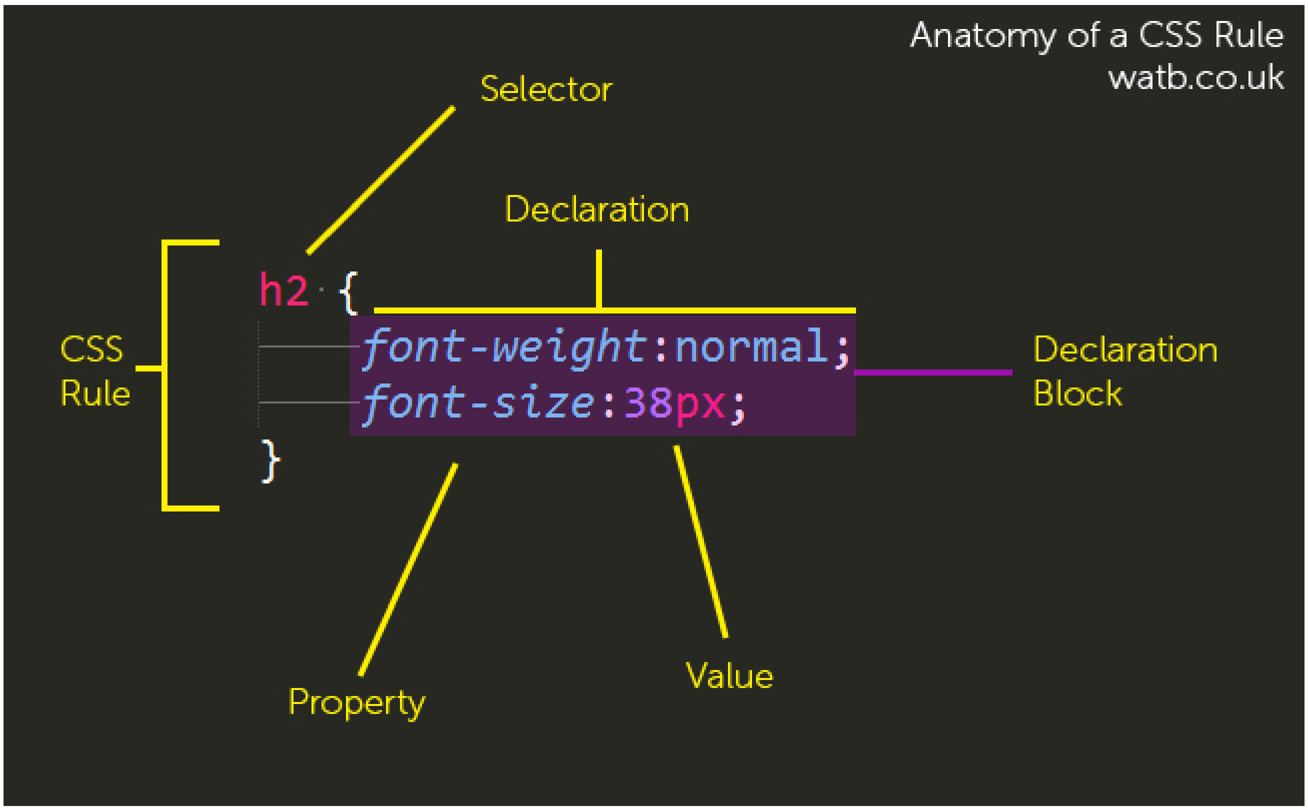
Week 3_2(en)

CSS Rule Structure and Selectors

> Now let's learn the syntax of CSS!

CSS Rule Structure



```css
h2 {
  font-weight:normal;
  font-size:38px;
}
```

This is how a typical CSS rule looks like.

- Selector: indicates the part we are going to apply the style to
- Declaration Block: within the `{}`, specify how we want to style the part we selected
- Declaration: consisted of property and its value in the form of `property: value;`
- We can have as many declarations inside a rule as we like.

Different Types of Selectors

CSS can select HTML elements by tag, class, or ID.
Let's learn about each of them.

1. Tag Selector

As we did earlier, we can use a tag name as a selector to apply the style to all the elements of that name on our page.
We already have a rule for the `<h1>` tag. Let's modify the rule and add another one for the `<p>` tag.

```css
h1 {
  color: red;
  font-size: 30px;
  text-align: center;
}

p {
  font-size: 20px;
  font-family: Arial;
}
```

You will see that the rule is applied to every `<p>` element.

2. Class Selector

CSS is not limited to selecting elements by tag name.
HTML elements can have more than just a tag name. They can also have attributes!
One common element is the `class` attribute, which we used last time.
We've learned that the `class` attribute can be used to style our page in a consistent way.
Let's add the following code to our `blog.css` file.

```css
.south-korea {
  text-transform: uppercase;
  color: blueviolet;
}
```

To select an HTML element by its class, a period must come in front of the class name.

It is possible to add more than one class name to an HTML element's `class` attribute.
For example, let's say we want to make the title of our article darkgreen and bold.
We can add two rules like below.

```css
.darkgreen {
  color: darkgreen;
}

.bold {
  font-weight: bold;
}
```

Then, we can include both of these classes to a single element.

```html
<h2 class="darkgreen bold">...</h2>
```

Like this, we can add multiple classes to an element's `class` attribute by separating them with a space.
This prevents us from writing a custom class for every style combination needed, and enables us to mix and match CSS classes to create many unique styles.

3. ID Selector

We can style a specific element uniquely(*regardless of classes applied to the element*) by using an `id` attribute.
While classes are meant to be reused over many elements, an ID is meant to style only one element.
Let's add an `id` attribute to our `<p>` tag in the `<footer>`.

```
<p id="corp-name">&copy; YURI Corp. </p>
```

Then let's add the following rule to our `.css` file.

```
#corp-name {
  font-family: cursive;
  color: rebeccapurple;
}
```

To select an element by its `id` attribute, we need to prepend the `id` name with a hashtag.

Specificity

Specificity is the order by which the browser decides which CSS styles will be displayed when 2 or more styles are competing with one another.
IDs are more specific than classes, and classes are more specific than tags.
For example, let's add a `class` attribute to our corporation name like below.

```
<p id="corp-name" class="corp-name">
```

Then let's try adding `color: grey;` to `p` rule.
Also, let's add the following rule.

```
.corp-name {
  color: red;
}
```

You will notice that the corporation name has not changed to grey nor red.
This is because the browser selects the more specific rule, which is the rule specified by the ID selector in this case.
Let's try erasing the `id` attribute of the `<p>` tag temporarily.
Now you can see the corporation name in red, which means the class selector is more specific than the tag selector.

> **Note**: A best practice is to style elements while using the **lowest degree of specificity**, so that if an element needs a new style, it is easy to override using a more specific selector.
> To make styles easy to edit, style with a tag selector if possible. If not, add a class selector. If that is not specific enough, then consider using an ID selector.

Using more than one selector

Chaining Selectors

We can require an HTML element to have two or more CSS selectors at the same time when writing CSS rules.
Let's add the following `class` attribute to `<p>` elements which contain the date of each article, and `<footer>` elements which contain the number of comments and likes.

```
class="highlight"
```

Then let's add the following rule to our CSS file.

```
p.highlight {
  background: yellow;
}
```

You will notice that the style has been applied to only the `<p>` elements with the `highlight` class.
We can combine multiple selectors(*chain* selectors) like above.

Nested Elements

We can also select elements that are nested within other elements.
For example, let's add a class to our first `<article>` tag.

```
<article class="first-article">
```

Then let's add the following rule to our CSS file.

```
.first-article p {
  color: darkblue;
}
```

You can see that only the `<p>` element nested inside the element that has `first-article` as its `class` attribute is affected by the style.
We can select nested `<p>` elements by adding `p` to the selector, separated by a space, resulting in `.first-article p`.

Selecting elements in this way can make our selectors even more specific by making sure they appear in the context we expect.

Chaining and Specificity

Adding more than one tag, class, or ID to a CSS selector **increases** the specificity of the CSS selector.
For example, we have both of the following rules in our CSS file.

```css
p {
  font-size: 20px;
  font-family: Arial;
  color: grey;
}

.first-article p {
  color: darkblue;
}
```

Although both of these rules define what a `p` element should look like, in our browser, we can check that the second style is applied instead of the first one in our first article.

Multiple Selectors

By using a comma, we can add CSS styles to multiple CSS selectors all at once.
For instance, let's change the `.corp-name` rule like below.

```css
.corp-name {
  color: darkgreen;
}
```

We can notice that the `.darkgreen` rule has the same declaration block as the `.corp-name` rule.
To keep the code concise, we can apply the same style to both like below.

```css
.corp-name, .darkgreen {
  color: darkgreen
}
```