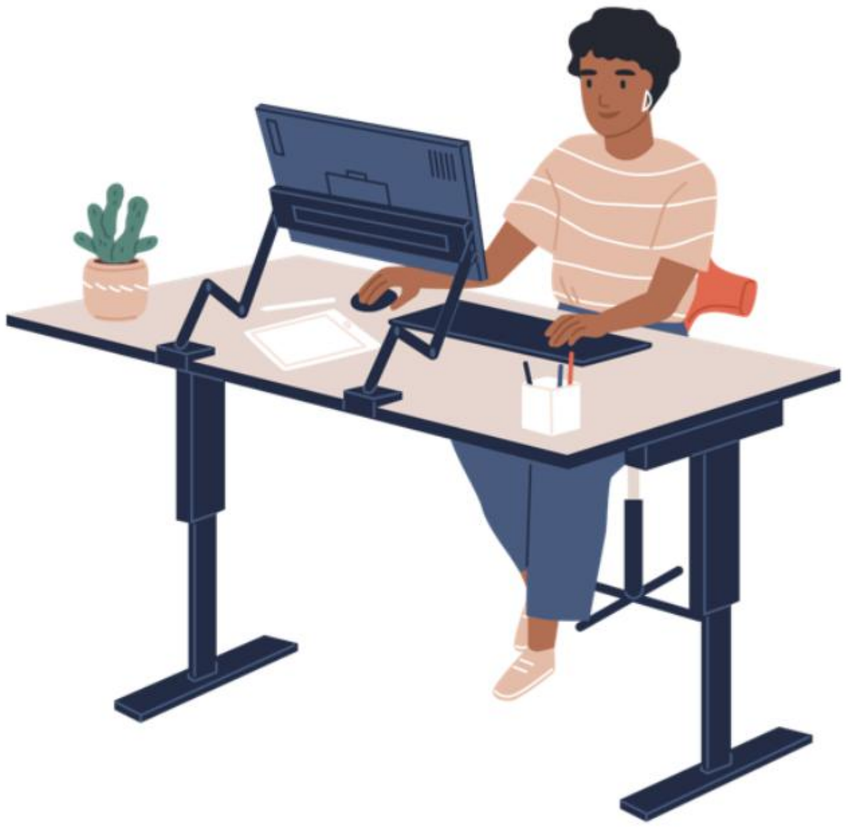
Slide Note

Learning Consolidation

Style a Web Page Using CSS Properties and CSS Box Model



Menu

00:01:00:00:25

01/ 25

Slide Note

In this sprint, you learned to:

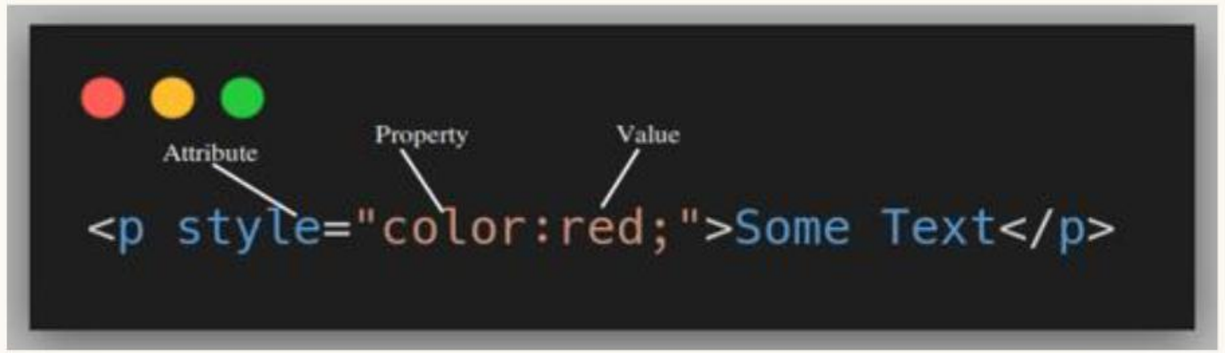
- Use CSS to style an HTML page
- Select HTML elements using CSS selectors
- Apply styles using CSS properties
- Create layouts by using the CSS box model
- Differentiate between the block and inline element

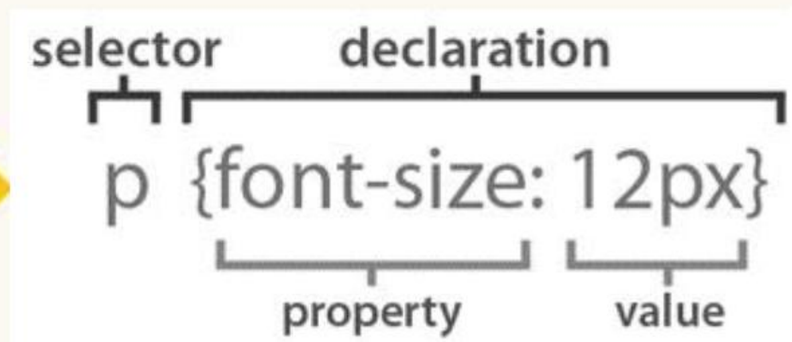
Slide Note

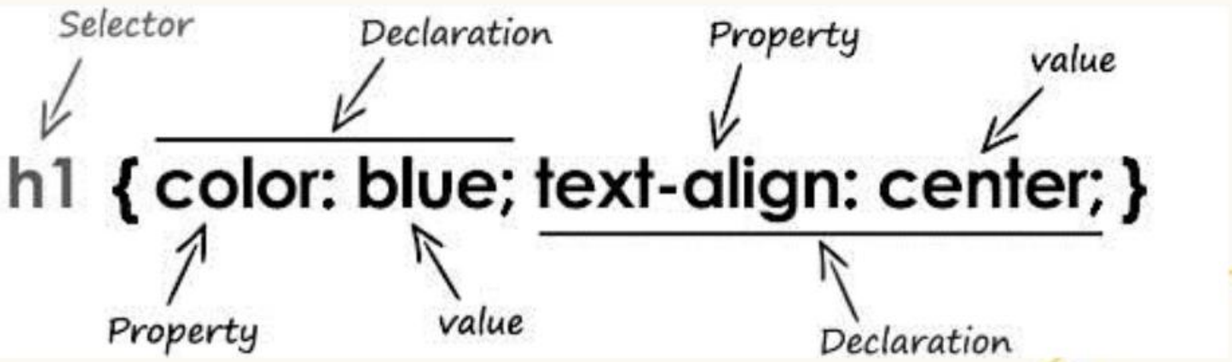
- A CSS rule set consists of a selector and a declaration block.
- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a property name and a value, separated by a colon.

What Is CSS (Cascading Style Sheet)?

CSS defines how HTML elements are to be displayed using certain rules.







Slide Note

Applying Styles to an HTML Page

- **Inline Style**

  Inline styles are placed directly inside an HTML element in the code using the "style" attribute.

- **Embedded or Internal Style**

  Internal styles are placed inside the head section of a particular web page via the style tag.

- **External Style**

  An external style sheet is a separate page which is then linked to the web page.

Slide Note

CSS Properties

- Background properties
- Font and text properties
- Positioning properties
- Styling links properties
- List-style properties
- Styling tables properties

Slide Note

CSS Selectors

Method	Description
Type selector	h1 { }
Universal selector	* { }
Class selector	.box { }
id selector	#unique { }
Attribute selector	a[title] { }
Pseudo-class selectors	p:first-child { }
Pseudo-element selectors	p::first-line { }
Descendant combinator	article p
Child combinator	article > p
Adjacent sibling combinator	h1 + p
General sibling combinator	h1 ~ p

Slide Note

Descendant Selector

The descendant selector matches all elements that are descendants of a specified element.

Child Selector

The child selector selects all elements that are the immediate children of a specified element.

Adjacent Sibling Selector

The adjacent sibling selector selects all elements that are the adjacent siblings of a specified element.

Sibling elements must have the same parent element, and "adjacent" means "immediately following".

General Sibling Selector

The general sibling selector selects all elements that are siblings of a specified element.

Combinator Selectors

A CSS selector can contain more than one simple selector. We can include a combinator between simple selectors.

- There are four different combinators in CSS3:
  - Descendant selector (space)
  - Child selector (>)
  - Adjacent sibling selector (+)
  - General sibling selector (~)

Slide Note

Combinator Selectors (cont'd)

```html
<!-- HTML Code -->
<section>
    <h2>
        This heading is a direct child of the
        section tag.
    </h2>
    <p>Follows the paragraph tag.</p>
    <article>
        <p>
            This paragraph tag is NOT a direct
             child of the section tag
        </p>
    </article>
    <p>
      This paragraph tag is also a child of t
      he section tag
    </p>
</section>
```

```css
/* CSS Code for child selector*/
section>p {
    color: green;
    font-family: sans-serif;
    font-weight: bold;
}
```

Slide Note

Self-Check

Five generic font-family values can be assigned to the font-family property. Three of them are provided below. Which are the other two?

- San-serif
- Cursive
- Monospace
- _____
- _____

Slide Note

Self-Check: Solution

Five generic font-family values can be assigned to the font-family property. Three of them are provided below. Which are the other two?

- San-serif
- Cursive
- Monospace
- **Serif**
- **Fantasy**

Slide Note

Self-Check

Which one of the following indicates the usage of the next sibling selector?

A. #content + p { text-align: center;}

B. #content > p { text-align: center;}

C. #content p { text-align: center;}

D. #content ~ p { text-align: center;}

Slide Note

Self-Check: Solution

Which one of the following indicates the usage of the next sibling selector?

A. **#content + p { text-align: center;}**

B. #content > p { text-align: center;}

C. #content p { text-align: center;}

D. #content ~ p { text-align: center;}

Explanation:

Option B is incorrect because the (>) symbol is used for the child selector.
Option C is incorrect because (space) is used for the descendant selector.
Option D is incorrect because (~) is used for the general sibling selector.

Slide Note

CSS Box Model

Slide Note

Open a website like wikipedia.com.

1. Right click on the "Contents" hyperlink in the browser window

2. Click inspect menu to open chrome dev tools.

3. Select the Computed tab at the top of the rightmost column.

4. Hover over the different properties of the logo's box. The corresponding space on the web page should be highlighted when you do this.

CSS Box Model

The CSS box model is a box that wraps around every HTML element. It consists of margins, borders, padding, and the actual content.

- **Margin:** This is the outermost layer that wraps the content, with the padding and border as the white space between this box and other elements.
- **Border**: Wraps the content and the padding, if any.
- **Padding**: Sits around the content as the white space; its size can be controlled using padding and related properties.
- **Content**: The area where your content is displayed.

Slide Note

Block Boxes and Inline Boxes

The characteristic of these boxes determines how they behave in terms of page flow and in relation to other boxes of the page.

Block Box	Inline Box
The box will break into a new line.	The box will not break into a new line.
Always take the full width available.	Takes only as much width as is necessary.
Width and height properties are respected	Width and height properties are not respected.
Examples: , <h1>, <p>	Example: , <img>

Slide Note

Cascade and Inheritance

The cascade, and the closely related concept of specificity, are mechanisms that control which rule would apply when there is such a conflict.

The concept of inheritance, which means that some CSS properties by default inherit values set on the current element's parent element, and some don't. This can also cause some unexpected behavior.

Slide Note

Self-Check

```
<div id="header">
    <h1>Main Heading</h1>
</div>
<div id="content">
    <h1>Hello!</h1>
    <p>
        Paragraph begins here...
    </p>
</div>
```

Choose the correct output.

A. h1.content { …. }

B. h1#content{ …. }

C. .content h1{ …. }

D. #content h1{ …. }

Slide Note

Self-Check: Solution

```
<div id="header">
    <h1>Main Heading</h1>
</div>
<div id="content">
    <h1>Hello!</h1>
    <p>
        Paragraph begins here...
    </p>
</div>
```

Choose the correct output.

A. h1.content { …. }

B. h1#content{ …. }

C. .content h1{ …. }

D. #content h1{ …. }

Explanation: ID selectors must be preceded with # and to select the child element inside the div, the combinator selector (space) must be used; hence, Option D is the correct choice.

Slide Note

When selectors have an equal specificity value, the latest rule is the one that counts.

When selectors have an unequal specificity value, the more specific rule is the one that counts.

Rules with more specific selectors have a greater specificity.

The last rule defined overrides any previous, conflicting rules.

The embedded style sheet has a greater specificity than other rules.

ID selectors have a higher specificity than attribute selectors.

You should always try to use IDs to increase the specificity.

A class selector beats any number of element selectors.

The universal selector and inherited selectors have a specificity of 0, 0, 0, 0.

You can calculate CSS specificity with the CSS Specificity Calculator.

Check this link for directly targeted elements:

https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity#directly_ta

CSS Specificity

- Specificity is the means by which browsers decide which CSS property values are the most relevant to an element and can be applied.
- It is based on the matching rules which are composed of different sorts of CSS selectors.
- It only applies when the same element is targeted by multiple declarations.
- As per CSS rules, directly targeted elements always take precedence over rules which an element inherits from its ancestor.

Slide Note

Self-Check

Answer the following questions:

Base HTML

```
<main id="content">
    <h1>The World's Best Code Editor</h1>
</main>
```

Which style will be applied to the h1 element?

```
h1 { background-color: red }
h1 { background-color: yellow }
```

Which Style will be applied to h1 element External or Embedded Style Sheet?

```
/* External Stylesheet */
#content h1 { background-color: red }
<!-- Embedded Stylesheet -->
<style>
#content h1 {
    background-color: yellow
}
</style>
```

Slide Note

Self-Check: Solution

- Question 1:
  Equal Specificity: Latest Rule counts
  H1 will be in yellow color

- Question 2:
  Embedded style is closer to the element to be styled.
  H1 will be in yellow color

Slide Note

Self-Check

Answer the following questions:

Base HTML

```
<div id="a">
  <h1>Main Heading<h1>
</div>
```

Which style will be applied to the h1 element?

```
div#a{background-color:green;}
#a{a{background-color:yellow;}
div[id=a]{background-color:blue;}
```

Base HTML

```
<h1 class="intro">
  Main Heading
</h1
```

Which Style will be applied to h1 element - class selector or element selector?

```
.intro{background-color:yellow;}
h1{background-color:red;}
```

Slide Note

Self-Check: Solution

- Question 1:
  ID selectors have higher specificity than attribute selectors
  H1 will be in green color

- Question 2:
  Class selectors overrides any number of element selectors
  H1 will be in yellow color

Slide Note

Exception

Overriding inline styles

Base HTML

```
<div class="foo" style="color:
red;">What color am I?</div>
```

CSS Code

```
/*External stylesheet*/
.foo[style*="color: red"] {
 color: firebrick !important;
}
```

Div will be in firebrick color

Overriding high specificity

Base HTML

```
<p class="myclass" id="myid">
   Paragraph content
</p>
```

CSS Code

```
.myclass {color: red;}
#myid {color: blue;}
p {color: green !important}
```

Paragraph will be in green color