7.6 Modifying CSS with JavaScript

Modifying an element's inline style

JavaScript can manipulate a webpage's CSS to dynamically alter the looks of a webpage. Ex: JavaScript can change a background color when a button is clicked, or change the visibility of an error message when an input field is left blank. The ***CSS Object Model*** (***CSSOM***) is a set of APIs that allow JavaScript to manipulate CSS properties of a webpage.

Every element in the DOM has a ***style*** property that holds the inline styles set on the element. The `style` object implements the CSSOM interface ***CSSStyleDeclaration***, which provides methods for accessing, modifying, and removing CSS properties:

- The ***getPropertyValue()*** method returns the value of an element's CSS property or an empty string if the property is not set. Ex: `elem.style.getPropertyValue("color")` gets the element's color value.
- The ***setProperty()*** method sets the value of an element's CSS property. Ex: `elem.style.setProperty("color", "blue")` sets the element's color to blue.
- The ***removeProperty()*** method removes an element's CSS property. Ex: `elem.style.removeProperty("color")` removes the element's color property.

The `style` CSS properties can alternatively be accessed and modified using JavaScript property names. Ex: `elem.style.color = "blue"` is equivalent to `elem.style.setProperty("color", "blue")`. CSS property names that have dashes are converted into property names that use camel case. Ex: `background-color` becomes the JavaScript property `backgroundColor`.

PARTICIPATION ACTIVITY 7.6.1: Modifying inline CSS style.

Start 2x speed

```
<div id="mydiv" style="width: 100px; height: 100px;
                       background-color: green">
</div>
```

```
let myDiv = document.getElementById("mydiv");
let width = myDiv.style.getPropertyValue("width");
width = parseInt(width) + 100;
myDiv.style.setProperty("width", width + "px");
myDiv.style.setProperty("background-color", "red");
myDiv.style.removeProperty("width");
```

element.style

~~width~~	~~"200px"~~
height	"100px"
background-color	"red"

1. The inline style properties make the a green 100px wide square.
2. The DOM maintains a list of the div's style properties and values.
3. The element.style.getPropertyValue() method returns the 100px width as a string.
4. 100px is added to width, and element.style.setProperty() sets the div's width to 200px. The browser automatically renders the div with the new width.
5. Changing the div's background color to red automatically renders the div red.
6. removeProperty() removes the width property, so the by default spans the entire browser width.

Feedback?

PARTICIPATION ACTIVITY 7.6.2: Modifying inline style.

Refer to the HTML and JavaScript below.

```
<span style="color: green">TEST</span>

<script>
let span = document.querySelector("span");
</script>
```

1) What does the code below output to the console?

```
console.log(span.style.getPropertyValue("color"));
```

- color
- (●) green
- rgb(0, 255, 0)

Correct

`getPropertyValue()` returns the inline CSS property value of `color`, which is set in the HTML: `style="color: green"`.

2) What does the code below output to the console?

```
console.log(span.style.getPropertyValue("width"));
```

- (●) An empty string
- The span's pixel width
- `false`

Correct

The span's `style` attribute does not set the `width` property. `getPropertyValue()` returns an empty string for unset properties.

3) What is needed to set the span's background color?

```
span.____;
```

- `setProperty("background-color", "lightblue")`
- `style.setProperty(background-color, lightblue)`
- ◉ `style.setProperty("background-color", "lightblue")`

Correct

`setProperty()` sets the span's `background-color` property to "lightblue".

4) What is equivalent to the following code?

```
span.style.setProperty("font-family", "Arial");
```

- `span.style.font-family = "Arial";`
- ◉ `span.style.fontFamily = "Arial";`
- `span.style.fontfamily = "Arial";`

Correct

`fontFamily` is the JavaScript property name for the CSS property `font-family`.

Feedback?

Modifying a stylesheet

The ***document.styleSheets*** object is a list of the stylesheets used in the webpage. Each stylesheet in `document.styleSheets` is a ***CSSStyleSheet*** object, which maintains a list of the stylesheet's CSS rules in the property called ***cssRules***. Two `CSSStyleSheet` methods allow CSS rules to be added or removed:

- The ***insertRule()*** method inserts a new rule into the stylesheet. Ex: `document.styleSheets[0].insertRule("p { color: blue; }")` inserts a new paragraph rule that makes the text color blue.
- The ***deleteRule()*** method deletes a rule at a given index number from the stylesheet. Ex: `document.styleSheets[0].deleteRule(0)` deletes the first CSS rule from the first stylesheet.

The CSS properties from a CSS rule are accessible from the rule's `style` property, which implements the `CSSStyleDeclaration` interface. So a rule's CSS properties can be accessed, modified, and removed with `getPropertyValue()`, `setProperty()`, and `removeProperty()`. Ex: `document.styleSheets[0].cssRules[0].style.setProperty("color", "blue")` sets the stylesheet's first rule's color to blue.

Security issue

For security reasons, some browsers like Chrome restrict the ***`cssRules`*** *property from being accessed by JavaScript when the stylesheet is loaded off the computer's disk. The JavaScript and stylesheet must be downloaded from a web server for* ***`cssRules`*** *to be accessible.*

PARTICIPATION ACTIVITY 7.6.3: Insert, modify, and delete CSS rules.

The webpage below displays a menu of food items with 3 buttons underneath:

1. Insert Rule button - Calls `insertRule()` to add a new paragraph rule that turns the menu items' font color blue.
2. Change Rule button - Calls `changeRule()` to change the paragraph rule's color to red.
3. Delete Rule button - Calls `deleteRule()` to delete the paragraph rule, which turns the font color back to green.

Click the three buttons in order to watch the font color change from green to blue, blue to red, and finally back to green.

Make the following modifications:

1. Add code to `insertRule()` that inserts the rule `.price { font-weight: bold; }` so the prices appear bold.
2. Add code to `changeRule()` that changes the `.price` rule to include the property `font-style` set to `italic` so the prices appear bold and italic.
3. Add code to `deleteRule()` that deletes the `.price` rule so the font weight and style returns to normal.

After making the modifications, click the 3 buttons in order to verify the price font changes as expected.

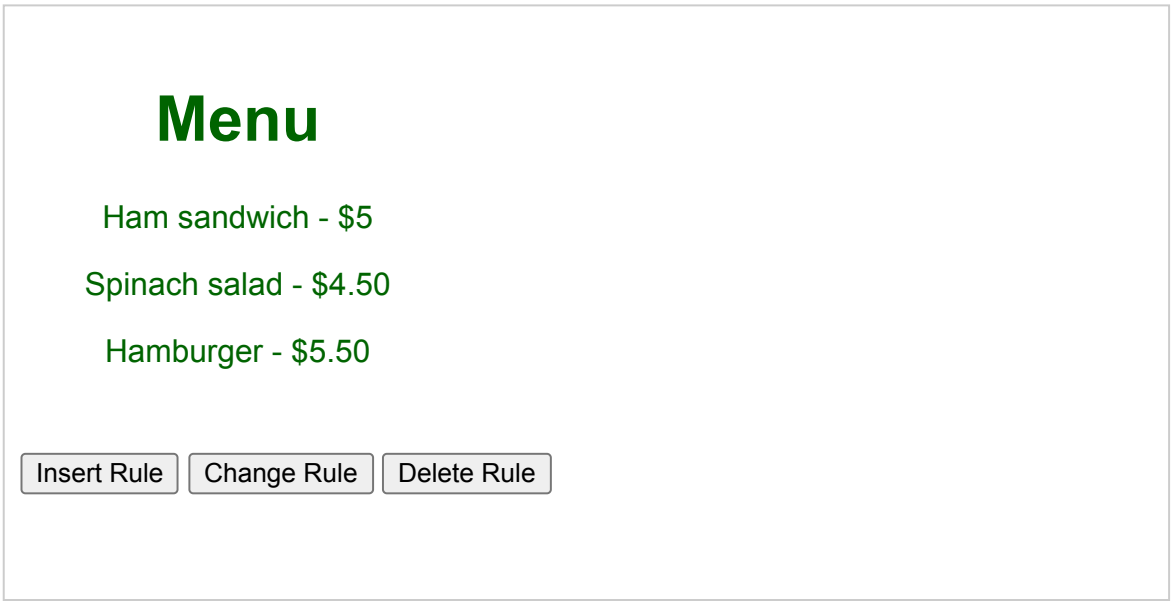
HTML | CSS | JavaScript

```
<body>
   <div id="menu">
      <h1>Menu</h1>
      <p>
         Ham sandwich - <span class="price">$5</span>
      </p>
      <p>
         Spinach salad - <span class="price">$4.50</span>
      </p>
      <p>
         Hamburger - <span class="price">$5.50</span>
      </p>
   </div>

   <p>
      <button id="insertRuleBtn">Insert Rule</button>
```

Render webpage Reset code

Your webpage



▼ View solution

Explain

```
--- START FILE: HTML ---



--- END FILE: HTML ---

--- START FILE: CSS ---

body  {
   color: darkgreen;
```

```
   font-family: Arial, Helvetica, sans-serif;
}

#menu {
   background-color: moccasin;
   width: 200px;
   text-align: center;
   padding: 10px;
   border-radius: 20px;
}


--- END FILE: CSS ---

--- START FILE: JavaScript ---

document.querySelector("#insertRuleBtn").addEventListener("cli
insertRule);
document.querySelector("#changeRuleBtn").addEventListener("cli
changeRule);
document.querySelector("#deleteRuleBtn").addEventListener("cli
deleteRule);

// Add paragraph rule
function insertRule() {
   let stylesheet = document.styleSheets[0];
   stylesheet.insertRule("p { color: blue; }");

   stylesheet.insertRule(".price { font-weight: bold; }");
}

// Change paragraph rule
function changeRule() {
   let stylesheet = document.styleSheets[0];
   for (let i = 0; i < stylesheet.cssRules.length; i++) {
      if (stylesheet.cssRules[i].selectorText === "p") {
         let style = stylesheet.cssRules[i].style;
         style.setProperty("color", "red");
      }
   }

   for (let i = 0; i < stylesheet.cssRules.length; i++) {
      if (stylesheet.cssRules[i].selectorText === ".price") {
         let style = stylesheet.cssRules[i].style;
         style.setProperty("font-style", "italic");
      }
   }
```

```
}

// Delete the paragraph rule
function deleteRule() {
   let stylesheet = document.styleSheets[0];
   for (let i = 0; i < stylesheet.cssRules.length; i++) {
      if (stylesheet.cssRules[i].selectorText === "p") {
         stylesheet.deleteRule(i);
      }
   }

   for (let i = 0; i < stylesheet.cssRules.length; i++) {
      if (stylesheet.cssRules[i].selectorText === ".price") {
         stylesheet.deleteRule(i);
      }
   }
}


--- END FILE: JavaScript ---
```

Feedback?

PARTICIPATION ACTIVITY | 7.6.4: Modifying stylesheet rules.

Refer to the HTML and CSS below.

```
<!DOCTYPE HTML>
<html lang="en">
   <head>
      <title>Funny Quotes</title>
      <link rel="stylesheet" href="styles.css">
   </head>
   <body>
      <blockquote
cite="https://www.brainyquote.com/quotes/tommy_cooper_189072">
         "I used to think I was indecisive, but now I'm not sure."
         - <span class="attribution">Tommy Cooper</span>
      </blockquote>
   </body>
</html>
```

```
/* styles.css */
body  {
   color: white;
   font-family: Arial, Helvetica, sans-serif;
}

blockquote {
   background-color: darkgreen;
   width: 200px;
   padding: 15px;
   border-radius: 5px;
}
```

1) What is the value of `document.styleSheets.length`?

- 0
- (•) 1
- 2

Correct

A single external stylesheet is imported with the `<link>` tag.

2) If the webpage is modified with the HTML below, what is the value of `document.styleSheets.length`?

```
<link rel="stylesheet"
href="styles.css">
<style>
div { color: brown; }
</style>
```

- 0
- 1
- (•) 2

Correct

The external stylesheet imported with `<link>` is `document.styleSheets[0]`, and the embedded stylesheet is `document.styleSheets[1]`.

3) What is the value of `document.styleSheets[0].cssRules.length`?

- 0
- 1
- (•) 2

Correct

The stylesheet declares 2 rules: `body` is `cssRules[0]`, and `blockquote` is `cssRules[1]`.

4) What is the font color of "Tommy Cooper" after the code below executes?

```
document.styleSheets[0].insertRule(".attribution
{ color: yellow; }");
```

- (•) yellow
- white
- black

Correct

The color is white before the code executes because of the `body` rule. The `<span>` tag around "Tommy Cooper" uses the `attribution` class, so adding

the `.attribution` rule changes the font color to yellow.

5) Assuming the first rule is `body`, what is the quote's font color after the code below executes?

```
document.styleSheets[0].deleteRule(0);
```

- yellow
- white
- black

Correct

Deleting the `body` rule makes the browser render the font using the browser's default color of black.

6) Assuming the second rule is `blockquote`, what is the quote's font color after the code below executes?

```
let quoteRule =
document.styleSheets[0].cssRules[1];
quoteRule.style.setProperty("color",
"orange");
```

- orange
- white
- black

Correct

The `color:orange` property is added to `blockquote`, which overrides the white color of the parent `<body>`.

Feedback?

Adding and removing classes

Using the CSSOM to manipulate CSS properties and stylesheets is useful and sometimes necessary, but mixing CSS with JavaScript code blurs the separation between a web application's presentation and functionality. *Good practice is to declare CSS classes that perform the styling and use JavaScript to add and remove classes to and from DOM nodes as needed.*

Every DOM node has a ***classList*** property that lists the classes assigned to the node. Ex: The div node created from `<div class="account warning">` has a `classList` with items "account" and "warning". Methods exist to add and remove `classList` items:

- The ***add()*** method adds a class to the node's `classList`. Ex: `elem.classList.add("mystyle")` adds the class `mystyle` to the element's list of classes.
- The ***remove()*** method removes a class from the node's `classList`. Ex: `elem.classList.remove("mystyle")` removes the class `mystyle` from the element's list of classes.

- The ***toggle()*** method adds the class to the node's `classList` if the class is not present. If the class is already present, the class is removed. Ex: `elem.classList.toggle("mystyle")` toggles the class `mystyle` on or off.

A DOM node's class list can also be modified directly using the ***className*** property, which is a space-delimited list of the classes assigned to the node. Ex: `elem.className = "cat adopted"` assigns the `cat` and `adopted` classes to the element and removes any previously assigned classes from the node. All classes assigned to `className` are also added to the node's `classList`. Adding and removing properties with `classList` is often easier than using `className`.

PARTICIPATION ACTIVITY 7.6.5: Add and remove classes.

The webpage below asks the user to enter a strong password that meets 3 criteria. When the user clicks the Submit button, the `isStrongPassword()` is called with the password entered.

- If the password does not meet all 3 criteria, `isStrongPassword()` returns `false` and an error message is displayed by removing the `hidden` class from the error message.
- If the password meets all 3 criteria, `isStrongPassword()` returns `true` and the `hidden` class is added to the error message to hide the error message.

Enter some passwords that cause the error message to be visible and then hidden. Ex: Enter "abc" and press Submit to see the error message, then "abcdef1" to hide the error message.

Modify the `submitBtnClick()` function to do the following:

1. If `isStrongPassword()` returns `true`, then remove the `error-textbox` class from the password text box.
2. If `isStrongPassword()` returns `false`, then add the `error-textbox` class to the password text box.

After making the modifications, verify the password text box is highlighted in red only when entering an invalid password.

For an extra challenge, add the `error` class to the criteria that is violated when an invalid password is entered. Ex: If the password is not long enough, add the `error` class to the first `<li>` so the item becomes red.

HTML | CSS | JavaScript

```
<body>
   <p>Choose a strong password that meets the following criteri
   </p>
   <ol>
      <li>At least 6 characters long.</li>
      <li>Contains at least 1 digit.</li>
      <li>Is not "password1".</li>
   </ol>
   <form>
      <label for="password">Password:</label>
      <input type="text" id="password">
      <span class="error hidden" id="errorMsg">Invalid passwor
      <div>
         <input type="button" id="submitBtn" value="Submit">
      </div>
   </form>
```

Render webpage Reset code

Your webpage

Choose a strong password that meets the following criteria:

1. At least 6 characters long.
2. Contains at least 1 digit.
3. Is not "password1".

Password: Submit

▼ View solution

Explain

```
--- START FILE: HTML ---



--- END FILE: HTML ---

--- START FILE: CSS ---
```

```
--- END FILE: CSS ---

--- START FILE: JavaScript ---

document.querySelector("#submitBtn").addEventListener("click",
submitBtnClick);

function isStrongPassword(password) {
   return password.length >= 6 && /\d/.test(password) &&
password !== "password1";
}

function submitBtnClick() {
   let password = document.querySelector("#password").value;
   if (isStrongPassword(password)) {

document.querySelector("#errorMsg").classList.add("hidden");

document.querySelector("#password").classList.remove("error-
textbox");
   } else {

document.querySelector("#errorMsg").classList.remove("hidden")

document.querySelector("#password").classList.add("error-
textbox");
   }
}


--- END FILE: JavaScript ---
```

Feedback?

PARTICIPATION ACTIVITY 7.6.6: Adding and removing classes.

Refer to the HTML, CSS, and JavaScript below:

```
<style>
   .important { background-color: yellow; }
   .complete { text-decoration: line-through; }
</style>
<body>
   <p>
     To-do list:
   </p>
   <ul>
      <li>Study for history exam</li>
      <li>Get groceries for dinner</li>
      <li>Volunteer at the children's center</li>
      <li>Vacuum and dust room</li>
   </ul>
</body>
```

```
// Add click callback to each <li>
const listItems = document.querySelectorAll("li");
for (let item of listItems) {
   item.addEventListener("click", listItemClick);
}

function listItemClick(e) {
   // Get clicked <li>
   let item = e.target;

}
```

Write the JavaScript code that is inserted into `listItemClick()` to perform the requested operation.

1) Add the `important` class to the clicked `<li>`.

```
item.classList.add("important");
```

Check Show answer

Correct

```
item.classList.add("important");
```

`add()` adds the class to the clicked `<li>`, making the `<li>` background turn yellow.

2) Remove the `complete` class from the clicked `<li>`.

```
item.classList.remove("complete");
```

Check Show answer

Correct

```
item.classList.remove("complete");
```

`remove()` removes the class from the `<li>` class list.

3) Toggle the `important` class on the clicked `<li>`.

```
item.classList.toggle("important");
```

Check Show answer

Correct

```
item.classList.toggle("important");
```

`toggle()` toggles the class, making the `<li>` background yellow or white.

Feedback?

Exploring further:

- An Introduction and Guide to the CSS Object Model (CSSOM) from css-tricks.com
- Using dynamic styling information from MDN
- Element.classList from MDN
- Working with the new CSS Typed Object Model from Google

CHALLENGE ACTIVITY 7.6.1: Modifying CSS with JavaScript.

530096.4000608.qx3zqy7

Jump to level 1

1 2 3

Complete the JavaScript code to add the alert class to the paragraph and remove the border class. **SHOW EXPECTED**

HTML JavaScript

```
1 const helloElem = document.querySelector("#helloMessage");
2
3 /* Your solution goes here */
4 helloElem.classList.add('alert');
5
6 // Remove the 'border' class
7 helloElem.classList.remove('border');
```

1 2 3

Check Next **Done**. Click any level to practice more. Completion is preserved.

✓ Testing value of class alert

Yours `Class alert is found.`

✓ Testing value of class border

Yours `Class border is removed.`

✔ Testing value of class comm

Yours `Class alert is found.`

Your webpage

Hello, Olympic contestants!

Feedback?

How was this section? Provide section feedback