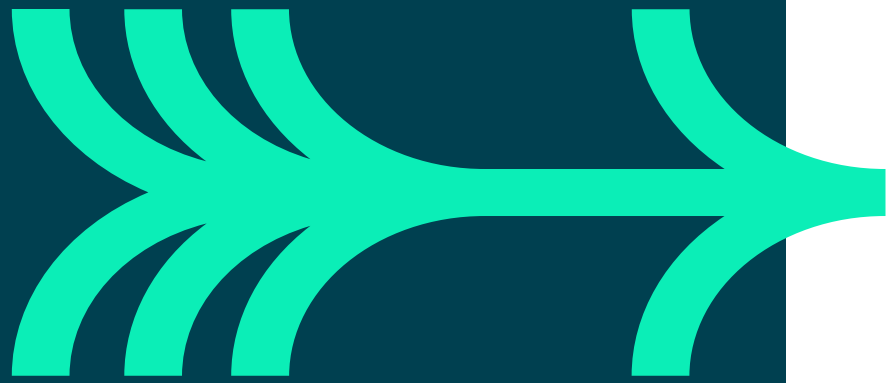
POWERING POTENTIAL

Manipulating styles

JavaScript Fundamentals



Introduction

HTML and the DOM

The style object

- Reading and setting CSS properties

CSS Classes and JavaScript

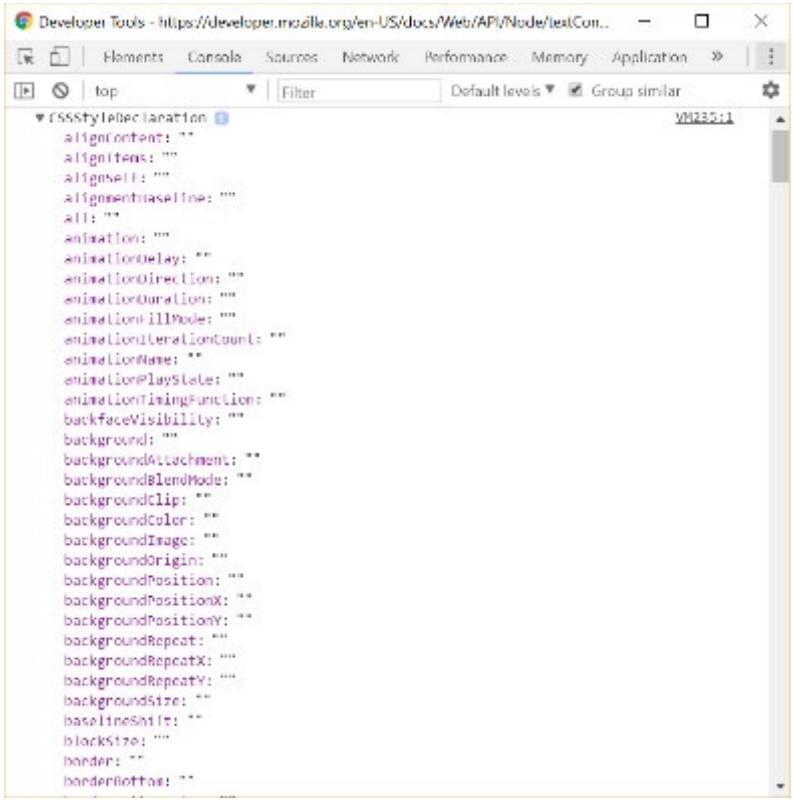
- The calculated style of an object
- Adding and removing classes

The style object

- Every rendered HTML object contains a style sub object

```
<p style="background-color: #b6ff00; border: 1px solid red" ></p>
```

- Use a **`console.dir()`** against the **`<p>`** declaration we will see
  - Each CSS rule is passed in as a parameter list
  - Applied to the appropriate property if understood
  - If not understood, a silent fail occurs
    - **Great for cross browser issues and new CSS features**



Reading CSS properties

- You can access an existing CSS property and assign it to a variable
  - CSS function can request any CSS property

```
let bgColor = document.querySelector('p:nth-child(2)').style.backgroundColor
```

- This gives you the element's style property, if there is one
  - If the property has been set via style or JavaScript, it returns a value
  - If by CSS class or ID you do not
- You receive the value that is part of the CSS Style Object, not necessarily what has actually been rendered

Setting multiple CSS Properties

- The style property can alter the CSS properties of an object
  - For writing properties to an existing object
  - You must ensure the element has rendered before you try to do this

```
document.querySelector('p:nth-child(1)').style.backgroundColor = '#dddddd';
document.querySelector('p:nth-child(1)').style.color = '#666666';
```

  - This could get repetitive. What about if we used Object.assign to help us out here?

```
let div = document.querySelector('div');

let styles = {
    backgroundColor: "pink",
    borderRadius: '5px',
    boxShadow: "5px 5px 5px deeppink"
}
Object.assign(div.style,styles);
```

CSS classes and JavaScript

- Classes are attached to a DOM object in a slightly different way

```
.myStyle {
    background-color: #4cff00;
    border: 1px solid red;
}
```

```
<p class="myStyle">test</p>
```

- Classes render as part of a DOM element
  - Represented as an array
  - Multiple classes can be added to an element
  - Their styles do not become party of the style property

Obtaining the calculated style of an object

- In most cases, we will read a CSS class from a style sheet
  - Use the **`window.getComputedStyle()`** method
  - Provides a read only final used values of the CSS
  - Returning a **`CSSStyleDeclaration`** object

```
let elem = document.querySelector('p:nth-child(1)');
let compStyle = getComputedStyle(elem).backgroundColor;
```

```
animationTimingFunction: "ease"
backfaceVisibility: "visible"
background: "rgb(76, 255, 0) none repeat scr
backgroundAttachment: "scroll"
backgroundBlendMode: "normal"
backgroundClip: "border-box"
backgroundColor: "rgb(76, 255, 0)"
backgroundImage: "none"
```

Adding and removing classes

- A class can be switched or added via JavaScript
  - Add or alter a class attribute
- Up to and including IE9
  - Removal must be done via string manipulation – it is quite tedious

```
const element = document.getElementById(elementID);
element.className = 'special';
```

```
const c = document.querySelector('#container');
c.className += ' div2';
```

- IE10 onwards
  - Can easily add and remove any class and modern browsers have the toggle method too!

```
const element = document.getElementById(elementID);
element.classList.add('special');
element.classList.remove('another-class');
```

QuickLab 10 – Manipulating Style with JavaScript

- Experiment with adding and removing styles on HTML elements

REVIEW

HTML to DOM rendering

Accessing the style object

- Reading style properties
- Setting style properties

Understanding CSS classes

- Obtaining the object computed style
- Applying classes via JavaScript