# 30 Days Of JavaScript: Document Object Model(DOM)

LinkedIn Follow @asabeneh

Author: Asabeneh Yetayeh

January, 2020

<< Day 20 | Day 22 >>

Thirty Days Of JavaScript

# Day 21

## Document Object Model (DOM) - Day 1

HTML document is structured as a JavaScript Object. Every HTML element has a different properties which can help to manipulate it. It is possible to get, create, append or remove HTML elements using JavaScript. Check the examples below. Selecting HTML element using JavaScript is similar to selecting using CSS. To select an HTML element, we use tag name, id, class name or other attributes.

### Getting Element

We can access already created element or elements using JavaScript. To access or get elements we use different methods. The code below has four *h1* elements. Let us see the different methods to access the *h1* elements.

```
<!DOCTYPE html>
  <html lang="en">
    <head>
      <title>Document Object Model</title>
    </head>
    <body>

     <h1 class='title' id='first-title'>First Title</h1>
     <h1 class='title' id='second-title'>Second Title</h1>
     <h1 class='title' id='third-title'>Third Title</h1>
     <h1></h1>

    </body>
  </html>
```

#### Getting elements by tag name

***getElementsByTagName()***:takes a tag name as a string parameter and this method returns an HTMLCollection object. An HTMLCollection is an array like object of HTML elements. The length property provides the size of the collection. Whenever we use this method we access the individual elements using index or after loop through each individual items. An HTMLCollection does not support all array methods therefore we should use regular for loop instead of forEach.

```
// syntax
document.getElementsByTagName('tagname')
```

```
const allTitles = document.getElementsByTagName('h1')

console.log(allTitles) //HTMLCollections
console.log(allTitles.length) // 4

for (let i = 0; i < allTitles.length; i++) {
  console.log(allTitles[i]) // prints each elements in the HTMLCollection
}
```

### Getting elements by class name

***getElementsByClassName()*** method returns an HTMLCollection object. An HTMLCollection is an array like list of HTML elements. The length property provides the size of the collection. It is possible to loop through all the HTMLCollection elements. See the example below

```
//syntax
document.getElementsByClassName('classname')
```

```
const allTitles = document.getElementsByClassName('title')

console.log(allTitles) //HTMLCollections
console.log(allTitles.length) // 4

for (let i = 0; i < allTitles.length; i++) {
  console.log(allTitles[i]) // prints each elements in the HTMLCollection
}
```

### Getting an element by id

***getElementsById()*** targets a single HTML element. We pass the id without # as an argument.

```
//syntax
document.getElementById('id')
```

```
let firstTitle = document.getElementById('first-title')
console.log(firstTitle) // <h1>First Title</h1>
```

### Getting elements by using querySelector methods

The *document.querySelector* method can select an HTML or HTML elements by tag name, by id or by class name.

***querySelector***: can be used to select HTML element by its tag name, id or class. If the tag name is used it selects only the first element.

```
let firstTitle = document.querySelector('h1') // select the first available h1 element
let firstTitle = document.querySelector('#first-title') // select id with first-title
let firstTitle = document.querySelector('.title') // select the first available element with c
```

***querySelectorAll***: can be used to select html elements by its tag name or class. It returns a nodeList which is an array like object which supports array methods. We can use ***for loop*** or ***forEach*** to loop through each nodeList elements.

```
const allTitles = document.querySelectorAll('h1') # selects all the available h1 elements in t

console.log(allTitles.length) // 4
for (let i = 0; i < allTitles.length; i++) {
  console.log(allTitles[i])
}

allTitles.forEach(title => console.log(title))
const allTitles = document.querySelectorAll('.title') // the same goes for selecting using cla
```

## Adding attribute

An attribute is added in the opening tag of HTML which gives additional information about the element. Common HTML attributes: id, class, src, style, href,disabled, title, alt. Lets add id and class for the fourth title.

```
const titles = document.querySelectorAll('h1')
titles[3].className = 'title'
titles[3].id = 'fourth-title'
```

### Adding attribute using setAttribute

The ***setAttribute()*** method set any html attribute. It takes two parameters the type of the attribute and the name of the attribute. Let's add class and id attribute for the fourth title.

```
const titles = document.querySelectorAll('h1')
titles[3].setAttribute('class', 'title')
titles[3].setAttribute('id', 'fourth-title')
```

### Adding attribute without setAttribute

We can use normal object setting method to set an attribute but this can not work for all elements. Some attributes are DOM object property and they can be set directly. For instance id and class

```
//another way to setting an attribute
titles[3].className = 'title'
titles[3].id = 'fourth-title'
```

#### Adding class using classList

The class list method is a good method to append additional class. It does not override the original class if a class exists rather it adds additional class for the element.

```
//another way to setting an attribute: append the class, doesn't over ride
titles[3].classList.add('title', 'header-title')
```

#### Removing class using remove

Similar to adding we can also remove class from an element. We can remove a specific class from an element.

```
//another way to setting an attribute: append the class, doesn't over ride
titles[3].classList.remove('title', 'header-title')
```

## Adding Text to HTML element

An HTML is a build block of an opening tag, a closing tag and a text content. We can add a text content using the property *textContent* or *innerHTML.

#### Adding Text content using textContent

The *textContent* property is used to add text to an HTML element.

```
const titles = document.querySelectorAll('h1')
titles[3].textContent = 'Fourth Title'
```

#### Adding Text Content using innerHTML

Most people get confused between *textContent* and *innerHTML*. *textContent* is meant to add text to an HTML element, however innerHTML can add a text or HTML element or elements as a child.

##### Text Content

We assign *textContent* HTML object property to a text

```
const titles = document.querySelectorAll('h1')
titles[3].textContent = 'Fourth Title'
```

##### Inner HTML

We use innerHTML property when we like to replace or a completely new children content to a parent element. It value we assign is going to be a string of HTML elements.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>JavaScript for Everyone:DOM</title>
  </head>
  <body>
    <div class="wrapper">
        <h1>Asabeneh Yetayeh challenges in 2020</h1>
        <h2>30DaysOfJavaScript Challenge</h2>
        <ul></ul>
    </div>
    <script>
    const lists = `
    <li>30DaysOfPython Challenge Done</li>
            <li>30DaysOfJavaScript Challenge Ongoing</li>
            <li>30DaysOfReact Challenge Coming</li>
            <li>30DaysOfFullStack Challenge Coming</li>
            <li>30DaysOfDataAnalysis Challenge Coming</li>
            <li>30DaysOfReactNative Challenge Coming</li>
            <li>30DaysOfMachineLearning Challenge Coming</li>`
  const ul = document.querySelector('ul')
  ul.innerHTML = lists
    </script>
  </body>
</html>
```

The innerHTML property can allow us also to remove all the children of a parent element. Instead of using removeChild() I would recommend the following method.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>JavaScript for Everyone:DOM</title>
  </head>
  <body>
    <div class="wrapper">
        <h1>Asabeneh Yetayeh challenges in 2020</h1>
        <h2>30DaysOfJavaScript Challenge</h2>
        <ul>
            <li>30DaysOfPython Challenge Done</li>
            <li>30DaysOfJavaScript Challenge Ongoing</li>
            <li>30DaysOfReact Challenge Coming</li>
            <li>30DaysOfFullStack Challenge Coming</li>
            <li>30DaysOfDataAnalysis Challenge Coming</li>
            <li>30DaysOfReactNative Challenge Coming</li>
            <li>30DaysOfMachineLearning Challenge Coming</li>
        </ul>
    </div>
    <script>
  const ul = document.querySelector('ul')
```

```
    ul.innerHTML = ''
      </script>
    </body>
  </html>
```

## Adding style

### Adding Style Color

Let us add some style to our titles. If the element has even index we give it green color else red.

```
const titles = document.querySelectorAll('h1')
titles.forEach((title, i) => {
  title.style.fontSize = '24px' // all titles will have 24px font size
  if (i % 2 === 0) {
    title.style.color = 'green'
  } else {
    title.style.color = 'red'
  }
})
```

### Adding Style Background Color

Let us add some style to our titles. If the element has even index we give it green color else red.

```
const titles = document.querySelectorAll('h1')
titles.forEach((title, i) => {
  title.style.fontSize = '24px' // all titles will have 24px font size
  if (i % 2 === 0) {
    title.style.backgroundColor = 'green'
  } else {
    title.style.backgroundColor = 'red'
  }
})
```

### Adding Style Font Size

Let us add some style to our titles. If the element has even index we give it 20px else 30px

```
const titles = document.querySelectorAll('h1')
titles.forEach((title, i) => {
  title.style.fontSize = '24px' // all titles will have 24px font size
  if (i % 2 === 0) {
    title.style.fontSize = '20px'
  } else {
    title.style.fontSize = '30px'
    }
  })
```

As you have notice, the properties of css when we use it in JavaScript is going to be a camelCase. The following CSS properties change from background-color to backgroundColor, font-size to fontSize, font-family to fontFamily, margin-bottom to marginBottom.

🌕 Now, you are fully charged with a super power, you have completed the most important and challenging part of the challenge and in general JavaScript. You learned DOM and now you have the capability to build and develop applications. Now do some exercises for your brain and for your muscle.

## Exercises

### Exercise: Level 1

1. Create an index.html file and put four p elements as above: Get the first paragraph by using ***document.querySelector(tagname)*** and tag name
2. Get each of the the paragraph using ***document.querySelector('#id')*** and by their id
3. Get all the p as nodeList using ***document.querySelectorAll(tagname)*** and by their tag name
4. Loop through the nodeList and get the text content of each paragraph
5. Set a text content to paragraph the fourth paragraph,***Fourth Paragraph***
6. Set id and class attribute for all the paragraphs using different attribute setting methods

### Exercise: Level 2

1. Change stye of each paragraph using JavaScript(eg. color, background, border, font-size, font-family)
2. Select all paragraphs and loop through each elements and give the first and third paragraph a color of green, and the second and the fourth paragraph a red color
3. Set text content, id and class to each paragraph

### Exercise: Level 3

#### DOM: Mini project 1

1. Develop the following application, use the following HTML elements to get started with. You will get the same code on starter folder. Apply all the styles and functionality using JavaScript only.

   - The year color is changing every 1 second
   - The date and time background color is changing every on seconds

- Completed challenge has background green
- Ongoing challenge has background yellow
- Coming challenges have background red

```html
<!-- index.html -->
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>JavaScript for Everyone:DOM</title>
  </head>
  <body>
    <div class="wrapper">
        <h1>Asabeneh Yetayeh challenges in 2020</h1>
        <h2>30DaysOfJavaScript Challenge</h2>
        <ul>
            <li>30DaysOfPython Challenge Done</li>
            <li>30DaysOfJavaScript Challenge Ongoing</li>
            <li>30DaysOfReact Challenge Coming</li>
            <li>30DaysOfFullStack Challenge Coming</li>
            <li>30DaysOfDataAnalysis Challenge Coming</li>
            <li>30DaysOfReactNative Challenge Coming</li>
            <li>30DaysOfMachineLearning Challenge Coming</li>
        </ul>
    </div>
  </body>
</html>
```

Project 1

Project 2

🎉 CONGRATULATIONS ! 🎉

<< Day 20 | Day 22 >>