JavaScript getElementsByClassName() Method

Summary: in this tutorial, you will learn how to use the JavaScript `getElementsByClassName()` method to select elements by class name.

Introduction to the getElementsByClassName() method

The `getElementsByClassName()` method returns an `HTMLCollection` of elements whose class names match one or more specified class names.

Here's the syntax of the `getElementsByClassName()` method:

```
getElementsByClassName(names)
```

In this syntax:

- `names` represents one or more class names to match. If you use multiple class names, you need to separate them by a space.

The `getElementsByClassName()` method returns a live `HTMLCollection` of the matched elements. This means that it will automatically update when the document changes.

If no element in the document matches the class names, the `getElementsByClassName()` method returns an empty `HTMLCollection []` .

The `getElementsByClassName()` method is available on both the `document` element and any other DOM elements.

When you call the `getElementsByClassName()` method on the `document` element, it will search the entire document and return the matched elements:

```
let elements = document.getElementsByClassName(names);
```

However, when you call the `getElementsByClassName()` method on a specific element, it will return the descendants of that particular element with the given class name:

```
let elements = element.getElementsByClassName(names);
```

JavaScript getElementsByClassName() method examples

Let's take some examples of using the `getElementsByClassName()` method.

Suppose you have the following HTML document:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>JavaScript getElementsByClassName</title>
</head>
<body>
    <header>
        <nav>
            <ul id="menu">
                <li class="item">HTML</li>
                <li class="item">CSS</li>
                <li class="item highlight">JavaScript</li>
                <li class="item">TypeScript</li>
            </ul>
        </nav>
        <h1>getElementsByClassName Demo</h1>
    </header>
    <section>
        <article>
            <h2 class="secondary">Example 1</h2>
        </article>
        <article>
            <h2 class="secondary">Example 2</h2>
        </article>
    </section>
```

```
</body>
</html>
```

1) Calling JavaScript getElementsByClassName() on an element example

The following example illustrates how to use the `getElementsByClassName()` method to select the `<li>` items which are the descendants of the `<ul>` element:

```
let menu = document.getElementById('menu');
let items = menu.getElementsByClassName('item');
console.log(items);

let data = [].map.call(items, (item) => item.textContent);
console.log(data);
```

Output:

```
HTMLCollection(4) [li.item, li.item, li.item.highlight, li.item]
['HTML', 'CSS', 'JavaScript', 'TypeScript']
```

How it works:

First, select the `<ul>` element with the class name `menu` using the `getElementById()` method:

```
let menu = document.getElementById('menu');
```

Next, select descendant elements of the ul element with the class name `item` using the `getElementsByClassName()` method:

```
let items = menu.getElementsByClassName('item');
```

Then, display the items in the console, which is an HTMLCollection that has four items:

```
console.log(items);
```

Output:

```
HTMLCollection(4) [li.item, li.item, li.item.highlight, li.item]
```

After that, create an array of items of the `HTMLCollection` using Array.of() method and returns an array of `textContent` of the items using the the `map()` method:

```
const data = Array.of(...items).map((item) => item.textContent);
```

Finally, display the `data` to the console:

```
console.log(data);
```

Output:

```
['HTML', 'CSS', 'JavaScript', 'TypeScript']
```

2) Calling JavaScript getElementsByClassName() on the document example

To select elements with the class `heading-secondary`, you use the following code:

```
const items = document.getElementsByClassName('secondary');
const data = Array.of(...items).map((item) => item.textContent);

console.log(data);
```

Output:

```
['Example 1', 'Example 2']
```

How it works.

First, select elements with the class secondary in the entire document:

```
const items = document.getElementsByClassName('secondary');
```

Second, create an array of `textContent` from the items in the `HTMLCollection` :

```
const data = Array.of(...items).map((item) => item.textContent);
```

Finally, display the `data` to the console:

```
console.log(data);
```

Summary

- Use the JavaScript `getElementsByClassName()` method to select the elements with one or more class names.

Quiz