Theory: ClassList

10 minutes 0 / 5 problems solved

The `classList` property is a `DOMTokenList` object that returns a list with the class name(s) of an element. The property is read-only, although you can manipulate the class of a specific element using the methods that we will learn in this topic.

§1. Syntax

Suppose we have an element with two classes `classOne` and `classTwo`.

```
<div class="classOne classTwo">classList</div>
```

To interact with the classes of an element we need to get that element first. We can use the `querySelector()` method, but you can use any other method as well.

```
let divElement = document.querySelector('.classOne');
console.log(divElement.classList); // ["classOne", "classTwo"]
```

Using only the property without any method will return a `DOMTokenList` with the names of the classes that the element has. As the return of the `classList` property is a list, we can manipulate it with the methods as an ordinary *JS* list.
Below we have a list of some methods of the property, let's see how we can use them properly.

- `add('classOne', 'classTwo')`: Adds one or more classes to the element.
- `remove('classOne', 'classTwo')`: Removes one or more classes from the element.
- `replace('classOne', 'classTwo')`: Replaces a class with another class.
- `toggle('class')`: Removes the class if the element contains the class, and adds it otherwise.
- `contains('class')`: Returns `true` if the element has the specified class, `false` otherwise.
- `length`: Returns the size of the list, i.e. the number of classes an element has.

§2. Add and remove method

We need to pass a ***string*** as a parameter, the ***string*** can not have spaces. If we want to pass more than one ***string*** we have to separate each one with a comma. We can use numbers too but it's not a good practice.

Let's add to our element a class named `classThree` using the `add()` method.

```
let divElement = document.querySelector('.classOne');
divElement.classList.add('classThree');

console.log(divElement.classList); // ["classOne", "classTwo", "classThree"]
```

If we want to add more than one class, we just have to separate each one with a comma, like in the example below.

```
divElement.classList.add('classThree', 'classFour');

console.log(divElement.classList); // ["classOne", "classTwo", "classThree", "clas
sFour"]
```

Now, if we want to remove these classes, we can use the `remove()` method. We can use as many classes as we want, just like with the `add()` method.

```
divElement.classList.remove('classThree', 'classFour');
console.log(divElement.classList); // ["classOne", "classTwo"]
```

Current topic:

ClassList

Topic depends on:

× DOM methods

Table of contents:

↑ ClassList
§1. Syntax
§2. Add and remove method
§3. Replace and toggle method
§4. Contains and length
§5. Conclusion
Feedback & Comments

§3. Replace and toggle method

To replace one class with another we can use the `replace()` method. This method takes two parameters. The first parameter is the class we want to remove and the second one is the class that we want to add.
Let's replace the class `classTwo` of our element with `classThree`.

```
let divElement = document.querySelector('.classOne');
console.log(divElement.classList); // ["classOne", "classTwo"]

divElement.classList.replace('classTwo', 'classThree');
console.log(divElement.classList); // ["classOne", "classThree"]
```

However, be careful: we need to pass two parameters and they cannot be empty. If we use an empty parameter, it will throw an `Uncaught DOMException`, or an `Uncaught TypeError` If we use only one parameter.

```
divElement.classList.replace('classTwo', ''); // Uncaught DOMException
divElement.classList.replace('classTwo'); // Uncaught TypeError
```

The `toggle()` method works like a switch. It removes the class if the element contains it and adds it if otherwise.

```
console.log(divElement.classList); // ["classOne", "classTwo"]

divElement.classList.toggle('classOne')
console.log(divElement.classList); // ["classTwo"]
```

In the example above the element contained the `classOne`, so it was removed. In the example below it's the opposite: the element does not contain the `classThree`, so it is added.

```
console.log(divElement.classList); // ["classOne", "classTwo"]

divElement.classList.toggle('classThree')

console.log(divElement.classList); // ["classOne", "classTwo", "classThree"]
```

§4. Contains and length

With the `contains()` method we can check if an element contains a certain class. Note that we can not manipulate the class of the element with this method. It only returns `true` if the element contains the class and `false` otherwise.

```
let divElement = document.querySelector('.classOne');
console.log(divElement.classList.contains('classOne')); // true
console.log(divElement.classList.contains('classThree')); // false
```

The `length` is not an exclusive method of the `classList` property, but since the property returns a list, we can use it. This method also does not change the classes of the elements: it returns the size of the list, or, in other words, how many classes that element has.

```
console.log(divElement.classList); // ["classOne", "classTwo"]
console.log(divElement.classList.length); // 2
```

§5. Conclusion

In this topic we've learned about the `classList` property that returns a list with the names of the classes of an element. We also learned about the `add()`, `remove()`, `replace()` and `toggle()` methods. We use these methods to manipulate the classes of an element. We've also learned about the `contains()` method used to check if an element contains a specific class, and the `length` method to check how many classes an element has.

Report a typo

5 users liked this theory. **0** didn't like it. **What about you?**

Start practicing

Comments (0) Hints (0) Useful links (0) Show discussion