perplexity

JavaScript `classList` Explained with Examples

The `classList` property in JavaScript provides an easy and powerful way to manipulate the list of CSS classes assigned to an HTML element. It returns a live `DOMTokenList` of the element's classes and offers several methods to add, remove, toggle, and check classes without directly handling the class attribute as a string[1] [2].

Key Features of `classList`

- Provides a convenient, readable API for working with CSS classes.
- Allows adding, removing, toggling, and replacing classes.
- Supports checking if a class is present.
- Enables adding/removing multiple classes at once.

Basic Usage Examples

Suppose you have the following HTML element:

```
<div id="myDiv" class="foo bar"></div>
```

Select the element in JavaScript:

```
const element = document.getElementById('myDiv');
```

Add a Class

```
element.classList.add('newClass');
// Now: <div id="myDiv" class="foo bar newClass"></div>
```

Adds the class `newClass` to the element[1] [3] [2].

Remove a Class

```
element.classList.remove('bar');
// Now: <div id="myDiv" class="foo newClass"></div>
```

Removes the class `bar` from the element[1] [2].

Toggle a Class

```
element.classList.toggle('active');
// Adds 'active' if not present, removes if present
```

Toggles the `active` class on or off[1] [2].

Check if a Class Exists

```
if (element.classList.contains('foo')) {
  // Do something if 'foo' is present
}
```

Returns `true` if the class exists, otherwise `false`[1] [2].

Replace a Class

```
element.classList.replace('foo', 'baz');
// Replaces 'foo' with 'baz'
```

Replaces an existing class with a new one[1] [2].

Add/Remove Multiple Classes

```
element.classList.add('first', 'second', 'third');
element.classList.remove('first', 'second');
```

You can add or remove several classes at once by passing multiple arguments[1] [2].

Get the Number of Classes

```
let count = element.classList.length;
```

Returns the number of classes on the element[1].

Get a Class by Index

```
let firstClass = element.classList.item(0);
```

Returns the class name at the specified index[1].

Practical Example: Toggle Dropdown

```
document.getElementById("myBtn").onclick = function() {
  document.getElementById("myDropdown").classList.toggle("show");
};
```

This toggles the `show` class on a dropdown menu when a button is clicked[1].

Summary Table of Common `classList` Methods

Method	Description	Example Usage
`.add(class1, ...)`	Adds one or more classes	`el.classList.add('a', 'b')`
`.remove(class1, ...)`	Removes one or more classes	`el.classList.remove('a', 'b')`
`.toggle(class, [force])`	Toggles a class on/off, or sets explicitly if `force`	`el.classList.toggle('a')`
`.contains(class)`	Checks if a class exists	`el.classList.contains('a')`
`.replace(old, new)`	Replaces an existing class with a new one	`el.classList.replace('a', 'b')`
`.length`	Number of classes	`el.classList.length`
`.item(index)`	Class name at a given index	`el.classList.item(0)`

Why Use `classList`?

- **Cleaner syntax**: No need to manually split/join class strings.
- **Safer**: Avoids accidental removal or duplication of class names.
- **More readable**: Methods clearly describe the intended action.

In summary:
The `classList` property is the modern, recommended way to manage CSS classes on DOM elements in JavaScript, making your code more robust and easier to maintain[1] [3] [2].

⁂

1. https://www.w3schools.com/jsref/prop_element_classlist.asp
2. https://developer.mozilla.org/en-US/docs/Web/API/Element/classList
3. https://www.javascripttutorial.net/javascript-dom/javascript-classlist/