How can I change an element's class with JavaScript?

Asked 16 years, 2 months ago Modified 4 months ago

Viewed 3.6m times

3441

How can I change the class of an HTML element in response to an `onclick` or any other events using JavaScript?

javascript html dom

Share Follow

edited Oct 27, 2020 at 5:52
Vibhor
115 ● 1 ● 14

asked Oct 12, 2008 at 20:06
Nathan Smith
37.5k ● 6 ● 30 ● 25

33 "The class attribute is mostly used to point to a class in a style sheet. However, it can also be used by a JavaScript (via the HTML DOM) to make changes to HTML elements with a specified class." - w3schools.com/tags/att_standard_class.asp – Triynko Apr 7, 2011 at 18:11

25 `element.setAttribute(name, value);` Replace `name` with `class`. Replace `value` with whatever name you have given the class, enclosed in quotes. This avoids

needing to delete the current class and adding a different one. This jsFiddle example shows full working code.
– Alan Wells May 18, 2014 at 4:59

5 For changing a class of HTML element with onClick use this code: `<input type='button' onclick='addNewClass(this)' value='Create' />` and in javascript section: `function addNewClass(elem){ elem.className="newClass"; }` Online
– Iman Bahrampour Aug 22, 2017 at 4:13

@Triynko - that link on w3schools has changed, looks like in September 2012. Here is that page on Archive.org from 12/Sep/2012: HTML class Attribute-w3schools. Here is the link for the replacement page on w3schools.com: HTML class Attribute-w3schools. – Kevin Fegan Oct 12, 2018 at 17:46

@ImanBahrampour That will obliterate any existing classes.
– Teepeemm Sep 17, 2021 at 2:53

34 Answers

Sorted by: Highest score (default)

1 2 Next

4653

Modern HTML5 Techniques for changing classes

Modern browsers have added **classList** which provides methods to make it easier to manipulate classes without needing a library:

```
document.getElementById("MyElement").classList.add('My
document.getElementById("MyElement").classList.remove(
if ( document.getElementById("MyElement").classList.co

document.getElementById("MyElement").classList.toggle(
```

Unfortunately, these do not work in Internet Explorer prior to v10, though there is a shim to add support for it to IE8 and IE9, available from this page. It is, though, getting more and more supported.

Simple cross-browser solution

The standard JavaScript way to select an element is using `document.getElementById("Id")`, which is what the following examples use - you can of course obtain elements in other ways, and in the right situation may simply use `this` instead - however, going into detail on this is beyond the scope of the answer.

To change all classes for an element:

To replace all existing classes with one or more new classes, set the className attribute:

```
document.getElementById("MyElement").className = "MyCl
```

(You can use a space-delimited list to apply multiple classes.)

To add an additional class to an element:

To add a class to an element, without removing/affecting existing values, append a space and the new classname, like so:

```
document.getElementById("MyElement").className += " My
```

To remove a class from an element:

To remove a single class to an element, without affecting other potential classes, a simple regex replace is required:

```
document.getElementById("MyElement").className =
   document.getElementById("MyElement").className.repl
      ( /(?:^|\s)MyClass(?!\S)/g , '' )
/* Code wrapped for readability - above is all one sta
```

An explanation of this regex is as follows:

```
(?:^|\s) # Match the start of the string or any single

MyClass  # The literal text for the classname to remov

(?!\S)   # Negative lookahead to verify the above is t
         # Ensures there is no non-space character fol
         # (i.e. must be the end of the string or spac
```

The `g` flag tells the replace to repeat as required, in case the class name has been added multiple times.

To check if a class is already applied to an element:

The same regex used above for removing a class can also be used as a check as to whether a particular class exists:

```
if (
document.getElementById("MyElement").className.match(/
```

Assigning these actions to onClick events:

Whilst it is possible to write JavaScript directly inside the HTML event attributes (such as `onClick="this.className+=' MyClass'"`) this is not recommended behavior. Especially on larger applications, more maintainable code is achieved by separating HTML markup from JavaScript interaction logic.

The first step to achieving this is by creating a function, and calling the function in the onClick attribute, for example:

```
<script type="text/javascript">
    function changeClass(){
        // Code examples from above
    }
</script>
...
<button onClick="changeClass()">My Button</button>
```

(It is not required to have this code in script tags, this is simply for the brevity of example, and including the JavaScript in a distinct file may be

more appropriate.)

The second step is to move the onClick event out of the HTML and into JavaScript, for example using addEventListener

```
<script type="text/javascript">
    function changeClass(){
        // Code examples from above
    }

    window.onload = function(){
        document.getElementById("MyElement").addEventL
changeClass);
    }
</script>
...
<button id="MyElement">My Button</button>
```

(Note that the window.onload part is required so that the contents of that function are executed *after* the HTML has finished loading - without this, the MyElement might not exist when the JavaScript code is called, so that line would fail.)

JavaScript Frameworks and Libraries

The above code is all in standard JavaScript, however, it is common practice to use either a framework or a library to simplify common tasks, as well as benefit from fixed bugs and edge cases that you might not think of when writing your code.

Whilst some people consider it overkill to add a ~50 KB framework for simply changing a class, if you are doing any substantial amount of JavaScript work or anything that might have unusual cross-browser behavior, it is well worth considering.

(Very roughly, a library is a set of tools designed for a specific task, whilst a framework generally contains multiple libraries and performs a complete set of duties.)

The examples above have been reproduced below using jQuery, probably the most commonly used JavaScript library (though there are others worth investigating too).

(Note that `$` here is the jQuery object.)

Changing Classes with jQuery:

```
$('#MyElement').addClass('MyClass');

$('#MyElement').removeClass('MyClass');

if ( $('#MyElement').hasClass('MyClass') )
```

In addition, jQuery provides a shortcut for adding a class if it doesn't apply, or removing a class that does:

```
$('#MyElement').toggleClass('MyClass');
```

Assigning a function to a click event with jQuery:

```
$('#MyElement').click(changeClass);
```

or, without needing an id:

```
$(':button:contains(My Button)').click(changeClass);
```

Share Follow

edited Aug 13 at 9:02

Ooker
2,896 ● 5 ● 37 ● 79

answered Oct 12, 2008 at 20:45

Peter Boughton
112k ● 32 ● 123 ● 177

---

122 Great answer Peter. One question... why is it **better** to do with with JQuery than Javascript? JQuery is great, but if this is all you need to do - what justifies including the entire JQuery libray instead of a few lines of JavaScript? – mattstuehler May 15, 2011 at 15:32

---

25 @mattstuehler 1) the phrase "better yet x" *often* means "better yet (you can) x". 2) To get to the heart of the matter, jQuery is designed to aid in accessing/manipulating the DOM, and very often if you need to do this sort of thing once you have to do it all over the place. – Barry May 24, 2011 at 16:46

---

33 One bug with this solution: When you click on your button multiple times, it will add the Class of " MyClass" to the element multiple times, rather than checking to see if it already exists. Thus you could end up with an HTML class attribute looking something like this: `class="button`

`MyClass MyClass MyClass"` – Web_Designer Sep 13, 2011 at 16:28

38 If you're trying to remove a class 'myClass' and you have a class 'prefix-myClass' the regex you gave above for removing a class will leave you with 'prefix-' in your className :O – jinglesthula Sep 15, 2011 at 5:26

19 Wow, three years and 183 upvotes and nobody spotted that until now. Thanks jinglesthula, I've corrected the regex so it wont incorrectly remove parts of class names. // I guess this is a good example of why a Framework (like jQuery) is worth using - bugs like this are caught and fixed sooner, and don't require changes to normal code. – Peter Boughton Sep 15, 2011 at 17:09

491

You could also just do:

```
document.getElementById('id').classList.add('class');
document.getElementById('id').classList.remove('class'
```

And to toggle a class (remove if exists else add it):

```
document.getElementById('id').classList.toggle('class'
```

Share Follow

edited Apr 4, 2020 at 19:27
johannchopin
14.8k ● 11 ● 62 ● 118

answered Aug 5, 2011 at 17:44
Tyilo
30k ● 41 ● 120 ● 201

70 I believe this is HTML5 dependent. – John Oct 27, 2011 at 17:16

13 You’ll need Eli Grey’s `classList` shim. – ELLIOTTCABLE Nov 14, 2011 at 14:34

16 Mozilla Developer Network states that it doesn't work, natively, in Internet Explorers less than 10. I find the statement to be true, in my testing. Apparently, the Eli Grey shim is required for Internet Explorer 8-9. Unfortunately, I couldn't find it on his site (even with searching). The shim is available on the Mozilla link. – doubleJ Oct 29, 2012 at 4:13

2 Here's the MDN Documentation for `classList`. – Ajedi32 Jun 15, 2015 at 15:56

5 atow "classList" has partial support in IE10+; no support for Opera Mini; else full support in remaining standard browsers: caniuse.com/#search=classlist – Nick Humphrey Jul 29, 2015 at 8:47

147

In one of my old projects that did not use jQuery, I built the following functions for adding, removing and checking if an element has a class:

```
function hasClass(ele, cls) {
    return ele.className.match(new RegExp('(\\s|^)' +
}

function addClass(ele, cls) {
    if (!hasClass(ele, cls))
        ele.className += " " + cls;
}

function removeClass(ele, cls) {
    if (hasClass(ele, cls)) {
        var reg = new RegExp('(\\s|^)' + cls + '(\\s|$
        ele.className = ele.className.replace(reg, ' '
    }
}
```

So, for example, if I want `onclick` to add some class to the button I can use this:

```
<script type="text/javascript">
    function changeClass(btn, cls) {
        if(!hasClass(btn, cls)) {
            addClass(btn, cls);
        }
    }
</script>
...
<button onclick="changeClass(this, "someClass")">My Bu
```

By now for sure it would just be better to use jQuery.

Share Follow

edited Aug 20, 2021 at 15:35

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered May 28, 2011 at 7:32

Andrew Orsich
53.7k ● 17 ● 141 ● 134

10 This is great for when your client doesn't let you use jQuery. (Cause you end up almost building your own library.) – Mike Jul 24, 2013 at 5:16

2 @Mike If the client doesn't let you use jQuery, could you not just go through and rebuild only the features you needed into your own library? – kfrncs Nov 18, 2013 at 5:04

6 @kfrncs Because I don't generally need that large of a framework. For the project I was thinking of, the only functions I needed were the 3 classname(has,add,remove) functions and the cookie(has, add, remove) functions. Everything else was either custom, or natively well supported. So everything together was then only 150 lines before minifying, including comments. – Mike Nov 18, 2013 at 19:17

This is my favorite solution for this. I use it everywhere. I believe it is the most elegant way to achieve adding and removing classes when your project does not already have another way of doing it. – WebWanderer Oct 17, 2017 at 21:36

It should be noted that after using `addClass` and `removeClass` on the same element, the element's className will contain an additional space. The className modifying line of `removeClass` should be updated to `ele.className = ele.className.replace(reg, ' ').trim().replace(/\s{2,}/g, ' ');`. This removes trailing whitespace left over and collapses multiple whitespaces into a single space in the className. – WebWanderer Oct 17, 2017 at 21:45

107

You can use `node.className` like so:

```
document.getElementById('foo').className = 'bar';
```

This should work in Internet Explorer 5.5 and up according to PPK.

Share Follow

edited Aug 20, 2021 at 15:23

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Oct 12, 2008 at 20:33

Eric Wendelin
44.3k ● 9 ● 72 ● 94

14 this would overwrite any and all other classes on the object... so it is not that simple. – Eric Sebasta Oct 9, 2015 at 20:33

62

Using pure JavaScript code:

```
function hasClass(ele, cls) {
    return ele.className.match(new RegExp('(\\s|^)' +
}

function addClass(ele, cls) {
    if (!this.hasClass(ele, cls)) ele.className += " "
}

function removeClass(ele, cls) {
    if (hasClass(ele, cls)) {
        var reg = new RegExp('(\\s|^)' + cls + '(\\s|$
        ele.className = ele.className.replace(reg, ' '
    }
}

function replaceClass(ele, oldClass, newClass){
    if(hasClass(ele, oldClass)){
        removeClass(ele, oldClass);
        addClass(ele, newClass);
    }
    return;
}

function toggleClass(ele, cls1, cls2){
    if(hasClass(ele, cls1)){
        replaceClass(ele, cls1, cls2);
    }else if(hasClass(ele, cls2)){
        replaceClass(ele, cls2, cls1);
    }else{
        addClass(ele, cls1);
    }
}
```

Share Follow

edited Jun 11, 2012 at 15:59

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Sep 20, 2011 at 15:26

PseudoNinja
2,856 ● 1 ● 29 ● 37

61

Wow, surprised there are so many overkill answers here...

```
<div class="firstClass" onclick="this.className='secon
```

Share Follow

edited Oct 19, 2014 at 23:47

answered Jan 5, 2012 at 19:14

Travis J
82.2k ● 42 ● 211 ● 279

15 I would say unobtrusive javascript is terrible practice for writing example code... – Gabe Apr 13, 2012 at 14:50

25 I would disagree, because I think example code should set a good example. – thomasrutter Mar 29, 2015 at 23:36

1 A good example should instruct and spark the imagination at the same time. It should not replace thought, but inspire it. – Anthony Rutledge Oct 31, 2015 at 4:01

5 The other answers aren't overkill, they also keep existing classes on the element. – gcampbell Jun 4, 2016 at 15:04

46

4 actions possible: Add, Remove, Check, and Toggle

Let's see multiple ways for each action.

1. Add class

Method 1: Best way to add a class in the modern browser is using `classList.add()` method of element.

- **Case 1**: Adding single class

```
function addClass() {
  let element = document.getElementById('id1');

  // adding class
  element.classList.add('beautify');
}
```

- **Case 2**: Adding multiple class

  To add multiple classes saperate classes by a comma in the `add()` method

```
function addClass() {
  let element = document.getElementById('id1');
  // adding multiple class
  element.classList.add('class1', 'class2', 'class
}
```

Method 2 - You can also add classes to HTML elements using `className` property.

- **Case 1**: Overwriting pre-existing classes When you assign a new class to the `className` property it overwrites the previous class.

```
function addClass() {
  let element = document.getElementById('id1');

  // adding multiple class
  element.className = 'beautify';
}
```

- **Case 2**: Adding class without overwrite Use `+=` operator for class not to overwrite previous classes. Also, add an extra space before the new class.

```
function addClass() {
  let element = document.getElementById('id1');

  // adding single multiple class
  element.className += ' beautify';
  // adding multiple classes
  element.className += ' class1 class2 class3';
}
```

---

2. Remove class

Method 1 - The best way to remove a class from an element is the `classList.remove()` method.

- **Case 1**: Remove single class

  Just pass the class name you want to remove from the element in the method.

  ```
  function removeClass() {
    let element = document.getElementById('id1');

    // removing class
    element.classList.remove('beautify');
  }
  ```

- **Case 2**: Remove multiple class

  Pass multiple classes separated by a comma.

  ```
  function removeClass() {
    let element = document.getElementById('id1');

    // removing class
    element.classList.remove('class1', 'class2', 'cl
  }
  ```

Method 2 - You can also remove class using `className` method.

- **Case 1**: Removing single class If the element has only 1 class and you want to remove it then just assign an empty string to the `className` method.

  ```
  function removeClass() {
    let element = document.getElementById('id1');

    // removing class
    element.className = '';
  }
  ```

- **Case 2**: Removing multiple class If the element has multiple classes first get all classes from the element using the `className` property and use replace method and replace desired classes with an empty string and finally assign it to element's `className` property.

  ```
  function removeClass() {
    let element = document.getElementById('id1');

    // removing class
    element.className = element.className.replace('c
  }
  ```

---

3. Checking class

To check if a class exists in the element you can simply use **`classList.contains()`** method. It returns `true` if the class exists else returns false.

```
function checkClass() {
  let element = document.getElementById('id1');

  // checking class
  if(element.classList.contains('beautify') {
      alert('Yes! class exists');
  }
}
```

---

4. Toggle class

To toggle a class use **`classList.toggle()`** method.

```
function toggleClass() {
    let element = document.getElementById('id1');

    // toggle class
    element.classList.toggle('beautify');
}
```

Share Follow

edited Dec 23, 2022 at 9:56
Glenn Antony Sheen
25 ● 1 ● 3

answered Sep 5, 2021 at 4:50
Satish Chandra Gupta
3,341 ● 1 ● 24 ● 23

43

This is working for me:

```
function setCSS(eleID) {
    var currTabElem = document.getElementById(eleID);

    currTabElem.setAttribute("class", "some_class_name
    currTabElem.setAttribute("className", "some_class_
}
```

Share Follow

edited Jun 11, 2012 at 16:01
Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Dec 8, 2011 at 9:36
Gopal Krishna Ranjan
878 ● 2 ● 12 ● 21

Excellent answer! Just left to add : Set for each CSS class name for selector to specify a style for a group of class elements – Roman Polen. May 8, 2012 at 11:19

This works for me on FF, but when I've tried to use el.className = "newStyle"; it didn't worked, why? – Lukasz 'Severiaan' Grela Jul 19, 2012 at 12:29

3 You can use `el.setAttribute('class', newClass)` or better `el.className = newClass`. But not `el.setAttribute('className', newClass)`. – Oriol Feb 11, 2015 at 15:54

29

As well you could extend HTMLElement object, in order to add methods to add, remove, toggle and check classes (gist):

```
HTMLElement = typeof(HTMLElement) != 'undefiend' ? HTM

HTMLElement.prototype.addClass = function(string) {
  if (!(string instanceof Array)) {
    string = string.split(' ');
  }
  for(var i = 0, len = string.length; i < len; ++i) {
    if (string[i] && !new RegExp('(\\s+|^)' + string[i
'(\\s+|$)').test(this.className)) {
      this.className = this.className.trim() + ' ' + s
    }
  }
}

HTMLElement.prototype.removeClass = function(string) {
  if (!(string instanceof Array)) {
    string = string.split(' ');
  }
  for(var i = 0, len = string.length; i < len; ++i) {
    this.className = this.className.replace(new RegExp
'(\\s+|$)'), ' ').trim();
  }
}
```

```
HTMLElement.prototype.toggleClass = function(string) {
  if (string) {
    if (new RegExp('(\\s+|^)' + string + '(\\s+|$)').t
      this.className = this.className.replace(new RegE
'(\\s+|$)'), ' ').trim();
    } else {
      this.className = this.className.trim() + ' ' + s
    }
  }
}

HTMLElement.prototype.hasClass = function(string) {
  return string && new RegExp('(\\s+|^)' + string +
'(\\s+|$)').test(this.className);
}
```

And then just use like this (on click will add or remove class):

```
document.getElementById('yourElementId').onclick = fun
  this.toggleClass('active');
}
```

Here is demo.

Share Follow

answered Apr 11, 2013 at 10:13

moka

23k ● 4 ● 52 ● 67

26

Just to add on information from another popular framework, Google Closures, see their dom/classes class:

```
goog.dom.classes.add(element, var_args)
goog.dom.classes.addRemove(element, classesToRemove, c

goog.dom.classes.remove(element, var_args)
```

One option for selecting the element is using goog.dom.query with a CSS 3 selector:

```
var myElement = goog.dom.query("#MyElement")[0];
```

Share Follow

edited Aug 20, 2021 at 15:36

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Nov 26, 2011 at 21:04

Ben Flynn
18.9k ● 21 ● 100 ● 143

22

Change an element's CSS class with JavaScript in ASP.NET:

```
Protected Sub Page_Load(ByVal sender As Object, ByVal 
Handles Me.Load
    If Not Page.IsPostBack Then
        lbSave.Attributes.Add("onmouseover", "this.cla
'LinkButtonStyle1'")
        lbSave.Attributes.Add("onmouseout", "this.clas
'LinkButtonStyle'")
        lbCancel.Attributes.Add("onmouseover", "this.c
'LinkButtonStyle1'")
        lbCancel.Attributes.Add("onmouseout", "this.cl
'LinkButtonStyle'")
    End If
End Sub
```

Share Follow

edited Jan 6, 2013 at 18:17

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered May 28, 2011 at 7:19

Hiren Kansara
241 ● 3 ● 4

22

A couple of minor notes and tweaks on the previous regexes:

You'll want to do it globally in case the class list has the class name more than once. And, you'll probably want to strip spaces from the ends of the class list and convert multiple spaces to one space to keep from getting runs of spaces. None of these things should be a problem if the only code dinking with the class names uses the regex below and removes a name before adding it. But. Well, who knows who might be dinking with the class name list.

This regex is case insensitive so that bugs in class names may show up before the code is used on a browser that doesn't care about case in class names.

```
var s = "testing   one   four  one  two";
var cls = "one";
var rg          = new RegExp("(^|\\s+)" + cls + "(\\s+
alert("[" + s.replace(rg, ' ') + "]");
var cls = "test";
var rg          = new RegExp("(^|\\s+)" + cls + "(\\s+
alert("[" + s.replace(rg, ' ') + "]");
var cls = "testing";
var rg          = new RegExp("(^|\\s+)" + cls + "(\\s+
alert("[" + s.replace(rg, ' ') + "]");
```

```
var cls = "tWo";
var rg          = new RegExp("(^|\\s+)" + cls + "(\\s+
alert("[" + s.replace(rg, ' ') + "]");
```

Share Follow

edited Nov 22, 2016 at 22:23
Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered May 2, 2012 at 4:59
Alex Robinson
609 ● 5 ● 4

19

The line

```
document.getElementById("MyElement").className =
document.getElementById("MyElement").className.replace
```

should be:

```
document.getElementById("MyElement").className =
document.getElementById("MyElement").className.replace
```

Share Follow

edited May 24, 2011 at 16:39
Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Dec 8, 2009 at 22:15
Eric Bailey
231 ● 2 ● 2

9 Incorrect. The RegEx is delimeted by the forward slashes. Adding quotes causes it to fail in IE, returning the string of the

class you're trying to remove rather than actually removing the class. – Dylan Aug 14, 2011 at 21:46

19

I would use jQuery and write something like this:

```
jQuery(function($) {
    $("#some-element").click(function() {
        $(this).toggleClass("clicked");
    });
});
```

This code adds a function to be called when an element of the id **some-element** is clicked. The function appends **clicked** to the element's class attribute if it's not already part of it, and removes it if it's there.

Yes, you do need to add a reference to the jQuery library in your page to use this code, but at least you can feel confident the most functions in the library would work on pretty much all the modern browsers, and it will save you time implementing your own code to do the same.

Share Follow

edited Aug 20, 2021 at 15:37
Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered May 5, 2012 at 2:46
shingokko
412 ● 3 ● 7

9 You only have to write `jQuery` in its long form once. `jQuery(function($) { });` makes `$` available inside

the function in all cases. – ThiefMaster Jun 13, 2012 at 14:38

17

Change an element's class in vanilla JavaScript with Internet Explorer 6 support

You may try to use the node `attributes` property to keep compatibility with old browsers, even Internet Explorer 6:

```
function getClassNode(element) {
  for (var i = element.attributes.length; i--;)
    if (element.attributes[i].nodeName === 'class')
      return element.attributes[i];
}

function removeClass(classNode, className) {
  var index, classList = classNode.value.split(' ');
  if ((index = classList.indexOf(className)) > -1) {
    classList.splice(index, 1);
    classNode.value = classList.join(' ');
  }
}

function hasClass(classNode, className) {
  return classNode.value.indexOf(className) > -1;
}

function addClass(classNode, className) {
  if (!hasClass(classNode, className))
    classNode.value += ' ' + className;
}

document.getElementById('message').addEventListener(
  var classNode = getClassNode(this);
  var className = hasClass(classNode, 'red') && 'blu
  removeClass(classNode, 'red');
  removeClass(classNode, 'blue');

  addClass(classNode, className);
})
```

```
.red {
  color: red;
}
.red:before {
  content: 'I am red! ';
}
.red:after {
  content: ' again';
}
.blue {
  color: blue;
}
.blue:before {
  content: 'I am blue! '
}
```

```
<span id="message" class="">Click me</span>
```

Run code snippet Expand snippet

Share Follow

edited Aug 20, 2021 at 15:47

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Nov 4, 2015 at 17:57

Eugene Tiurin
4,099 ● 4 ● 34 ● 32

14

Here's my version, fully working:

```
function addHTMLClass(item, classname) {
    var obj = item
    if (typeof item=="string") {
        obj = document.getElementById(item)
    }
    obj.className += " " + classname
}

function removeHTMLClass(item, classname) {
    var obj = item
    if (typeof item=="string") {
        obj = document.getElementById(item)
    }
    var classes = ""+obj.className
    while (classes.indexOf(classname)>-1) {
        classes = classes.replace (classname, "")
    }
    obj.className = classes
}
```

Usage:

```
<tr onmouseover='addHTMLClass(this,"clsSelected")'
onmouseout='removeHTMLClass(this,"clsSelected")' >
```

Share Follow

edited Jan 6, 2013 at 18:20

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Oct 17, 2012 at 12:21

alfred
1,018 ● 10 ● 13

5 That will break class `foobar` if you try to remove class `foo`. The JS in the intrinsic event handler attributes was broken before being edited. The 4 year old accepted answer is much better. – Quentin Oct 17, 2012 at 12:25

2 thanks, i fixed it (not the prefix problem). it's the old accepted answer that have a bug with the regexp. – alfred Oct 17, 2012 at 12:27

The code still have the `foobar` problem. See the test here – rosell.dk Oct 17, 2016 at 12:11

13

Here's a toggleClass to toggle/add/remove a class on an element:

```
// If newState is provided add/remove theClass accordi
theClass
function toggleClass(elem, theClass, newState) {
    var matchRegExp = new RegExp('(?:^|\\s)' + theClas
    var add=(arguments.length>2 ? newState : (elem.cla
== null));

    elem.className=elem.className.replace(matchRegExp,
    if (add) elem.className += ' ' + theClass;
}
```

See the JSFiddle.

Also see my answer here for creating a new class dynamically.

Share Follow

edited Aug 20, 2021 at 15:39

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Oct 28, 2015 at 7:00

kofifus
19.2k ● 21 ● 115 ● 183

11

I use the following vanilla JavaScript functions in my code. They use regular expressions and `indexOf` but do not require quoting special characters in regular expressions.

```
function addClass(el, cn) {
    var c0 = (" " + el.className + " ").replace(/\s+/g
        c1 = (" " + cn + " ").replace(/\s+/g, " ");
    if (c0.indexOf(c1) < 0) {
        el.className = (c0 + c1).replace(/\s+/g, " ").
    }
}

function delClass(el, cn) {
    var c0 = (" " + el.className + " ").replace(/\s+/g
        c1 = (" " + cn + " ").replace(/\s+/g, " ");
    if (c0.indexOf(c1) >= 0) {
        el.className = c0.replace(c1, " ").replace(/\s
$/g, "");
    }
}
```

You can also use element.classList in modern browsers.

Share Follow

edited Jan 18, 2014 at 9:37

answered Jan 18, 2014 at 9:28

Salman Arshad
272k ● 84 ● 441 ● 533

9

The OP question was *How can I change an element's class with JavaScript?*

Modern browsers allow you to do this **with one line of JavaScript**:

```
document.getElementById('id').classList.replace('span1', 'span2')
```

The `classList` attribute provides a DOMTokenList which has a variety of methods. You can operate on an element's classList using simple manipulations like *add()*, *remove()* or *replace()*. Or get very sophisticated and manipulate classes like you would an object or Map with *keys()*, *values()*, and *entries()*.

Peter Boughton's answer is a great one, but it's now over a decade old. All modern browsers now support DOMTokenList - see https://caniuse.com/#search=classList and even Internet Explorer 11 supports some DOMTokenList methods.

Share Follow

edited Aug 20, 2021 at 16:07

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered May 3, 2020 at 23:05

timbo
14.2k ● 9 ● 53 ● 74

It's amazing how many unnecessary regex can be found on this page, just to emulate the `classList.replace` command ;-) – David Nov 12 at 10:50

7

THE OPTIONS.

Here is a little style vs. classList examples to get you to see what are the options you have available and how to use `classList` to do what you want.

`style` vs. `classList`

The difference between `style` and `classList` is that with `style` you're adding to the style properties of the element, but `classList` is kinda controlling the class(es) of the element (`add`, `remove`, `toggle`, `contain`), I will show you how to use the `add` and `remove` method since those are the popular ones.

Style Example

If you want to add `margin-top` property into an element, you would do in such:

```
// Get the Element
const el = document.querySelector('#element');

// Add CSS property
el.style.margintop = "0px"
el.style.margintop = "25px" // This would add a 25px t
```

---

classList Example

Let say we have a `<div class="class-a class-b">`, in this case, we have 2 classes added to our div element already, `class-a` and `class-b`, and we want to control what classes **`remove`** and what class to **`add`**. This is where `classList` becomes handy.

Remove `class-b`

```
// Get the Element
const el = document.querySelector('#element');

// Remove class-b style from the element
el.classList.remove("class-b")
```

Add `class-c`

```
// Get the Element
const el = document.querySelector('#element');

// Add class-b style from the element
el.classList.add("class-c")
```

Share Follow

edited Jun 20, 2020 at 9:12

Community Bot
1 ● 1

answered Aug 22, 2019 at 18:28

Brian Nezhad
6,258 ● 10 ● 47 ● 70

Note that for using 'style' you might have to change your content security policy. – David Nov 11 at 21:40

6

Try:

```
element.className='second'
```

Show code snippet

Share Follow

edited Aug 20, 2021 at 15:55

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Apr 14, 2019 at 10:51

Kamil Kiełczewski
91.9k ● 34 ● 394 ● 370

6

For IE v6-9 (in which `classList` is not supported and you don't want to use polyfills):

```
var elem = document.getElementById('some-id');
```

```
// don't forget the extra space before the class name
var classList = elem.getAttribute('class') + ' other-c

elem.setAttribute('class', classList);
```

Share Follow

answered Sep 10, 2021 at 11:57

gomn
131 ●1 ●9

5

OK, I think in this case you should use jQuery or write your own Methods in pure JavaScript. My preference is adding my own methods rather than injecting all jQuery to my application if I don't need that for other reasons.

I'd like to do something like below as methods to my JavaScript framework to add few functionalities which handle adding classes, deleting classes, etc. Similar to jQuery, this is fully supported in IE9+. Also my code is written in ES6, so you need to make sure your browser support it or you using something like Babel, otherwise need to use ES5 syntax in your code. Also in this way, we finding element via ID, which means the element needs to have an ID to be selected:

```
// Simple JavaScript utilities for class management in
var classUtil = {

  addClass: (id, cl) => {
    document.getElementById(id).classList.add(cl);
  },

  removeClass: (id, cl) => {
    document.getElementById(id).classList.remove(cl);
  },
  hasClass: (id, cl) => {
    return document.getElementById(id).classList.conta
  },

  toggleClass: (id, cl) => {
    document.getElementById(id).classList.toggle(cl);
  }

}
```

And you can simply use them as below. Imagine your element has id of 'id' and class of 'class'. Make sure you pass them as a string. You can use the utility as below:

```
classUtil.addClass('myId', 'myClass');
classUtil.removeClass('myId', 'myClass');
classUtil.hasClass('myId', 'myClass');
classUtil.toggleClass('myId', 'myClass');
```

Share Follow

edited Aug 20, 2021 at 15:44
Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered May 29, 2017 at 13:29
Alireza
105k ● 27 ● 277 ● 173

4

`classList` DOM API:

A very convenient manner of adding and removing classes is the `classList` DOM API. This API allows us to select all classes of a specific DOM element in order to modify the list using JavaScript. For example:

```
const el = document.getElementById("main");
console.log(el.classList);
```

```
<div class="content wrapper animated" id="main"></di
```

Run code snippet Expand snippet

We can observe in the log that we are getting back an object with not only the classes of the element, but also many auxiliary methods and properties. This object inherits from the interface **DOMTokenList**, an interface which is used in the DOM to represent a set of space separated tokens (like classes).

Example:

```
const el = document.getElementById('container');

function addClass () {
    el.classList.add('newclass');
}


function replaceClass () {
    el.classList.replace('foo', 'newFoo');
}


function removeClass () {
    el.classList.remove('bar');
}
```

```
button{
  margin: 20px;
}

.foo{
  color: red;
}

.newFoo {
  color: blue;
}

.bar{
  background-color: powderblue;
}

.newclass{
  border: 2px solid green;
}
```

```
<div class="foo bar" id="container">
  "Sed ut perspiciatis unde omnis
  iste natus error sit voluptatem accusantium dolore
  totam rem aperiam, eaque ipsa quae ab illo invento
  quasi architecto beatae vitae dicta sunt explicabo
  voluptatem quia voluptas
 </div>

<button onclick="addClass()">AddClass</button>

<button onclick="replaceClass()">ReplaceClass</butto

<button onclick="removeClass()">removeClass</button>
```

Run code snippet Expand snippet

Share Follow

edited Aug 20, 2021 at 15:50

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Sep 21, 2018 at 9:23

Willem van der Veen
36.4k ● 18 ● 204 ● 176

4

Yes, there are many ways to do this. In ES6 syntax we can achieve easily. Use this code toggle add and remove class.

```
const tabs=document.querySelectorAll('.menu li');

for(let tab of tabs){

  tab.onclick = function(){

    let activetab = document.querySelectorAll('li.ac

    activetab[0].classList.remove('active')

    tab.classList.add('active');
  }

}
```

```
body {
    padding: 20px;
    font-family: sans-serif;
}
```

```css
ul {
    margin: 20px 0;
    list-style: none;
}

li {
    background: #dfdfdf;
    padding: 10px;
    margin: 6px 0;
    cursor: pointer;
}

li.active {
    background: #2794c7;
    font-weight: bold;
    color: #ffffff;
}
```

```html
<i>Please click an item:</i>

<ul class="menu">
  <li class="active"><span>Three</span></li>
  <li><span>Two</span></li>
  <li><span>One</span></li>
</ul>
```

Run code snippet Expand snippet

Share Follow

edited Aug 20, 2021 at 15:51

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Feb 6, 2019 at 14:02

Danish Khan
339 ● 2 ● 9

What do you mean by *"Use this code toggle add and remove class."*? It seems incomprehensible. E.g., is some punctuation missing? Please respond by [editing (changing) your answer](), not here in comments (***without*** "Edit:", "Update:", or similar - the answer should appear as if it was written today). – Peter Mortensen Aug 20, 2021 at 15:53

3

Just thought I'd throw this in:

```
function inArray(val, ary){
  for(var i=0,l=ary.length; i<l; i++){
    if(ary[i] === val){
      return true;
    }
  }
  return false;
}
function removeClassName(classNameS, fromElement){
  var x = classNameS.split(/\s/), s = fromElement.clas
[];
  for(var i=0,l=s.length; i<l; i++){
    if(!iA(s[i], x))r.push(s[i]);
  }
  fromElement.className = r.join(' ');
}
function addClassName(classNameS, toElement){
  var s = toElement.className.split(/\s/);
  s.push(c); toElement.className = s.join(' ');
}
```

Share Follow

answered Mar 20, 2015 at 2:35

StackSlave
10.6k ● 2 ● 19 ● 35

3

Just use `myElement.classList="new-class"` unless you need to maintain other existing classes in which case you can use the `classList.add, .remove` methods.

```
var doc = document;
var divOne = doc.getElementById("one");
var goButton = doc.getElementById("go");

goButton.addEventListener("click", function() {
  divOne.classList="blue";
});
```

```
div{
  min-height: 48px;
  min-width: 48px;
}
.bordered{
  border: 1px solid black;
}
.green{
  background: green;
}
.blue{
  background: blue;
}
```

```
<button id="go">Change Class</button>

<div id="one" class="bordered green">

</div>
```

Run code snippet   Expand snippet

Share Follow

edited Aug 20, 2021 at 15:42

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Apr 21, 2017 at 2:50

Ronnie Smith
18.5k ● 7 ● 88 ● 97

2

TL;DR:

```
document.getElementById('id').className = 'class'
```

OR

```
document.getElementById('id').classList.add('class');
document.getElementById('id').classList.remove('class'
```

That's it.

And, if you really want to know the why and educate yourself then I suggest reading Peter Boughton's answer. It's perfect.

Note:

This is possible with (*document* or *event*):

- `getElementById()`
- `getElementsByClassName()`
- `querySelector()`

- `querySelectorAll()`

Share Follow

edited Aug 20, 2021 at 15:55

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Feb 22, 2019 at 10:22

tfont
11.2k ● 7 ● 58 ● 52

2

```
function classed(el, class_name, add_class) {
  const re = new RegExp("(?:^|\\s)" + class_name + "(?
  if (add_class && !el.className.match(re)) el.classNa
  else if (!add_class) el.className = el.className.rep
}
```

Using Peter Boughton's answer, here is a simple cross-browser function to add and remove class.

Add class:

```
classed(document.getElementById("denis"), "active", tr
```

Remove class:

```
classed(document.getElementById("denis"), "active", fa
```

Share Follow

edited Aug 20, 2021 at 15:57

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Sep 4, 2019 at 15:32

donatso

108 ● 3

2

There is a property, **className**, in JavaScript to change the name of the class of an HTML element. The existing class value will be replaced with the new one, that you have assigned in className.

```
<!DOCTYPE html>
<html>
<head>
<title>How can I change the class of an HTML element i
<link rel="stylesheet" href="https://cdnjs.cloudflare.
awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
<h1 align="center"><i class="fa fa-home" id="icon"></i

<center><button id="change-class">Change Class</button

<script>
var change_class = document.getElementById("change-cla
change_class.onclick = function()
{
    var icon=document.getElementById("icon");
    icon.className = "fa fa-gear";
}
</script>
</body>
</html>
```

Credit - *How To Change Class Name of an HTML Element in JavaScript*

Share Follow

edited Aug 20, 2021 at 16:04

Peter Mortensen
31.6k ● 22 ● 109 ● 133

answered Mar 20, 2020 at 12:25

Jai Prakash
59 ● 3

1 2 Next

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.