Strip HTML from Text JavaScript

Asked 10 years, 3 months ago Active 2 months ago Viewed 550k times

570

167

Is there an easy way to take a string of html in JavaScript and strip out the html?

javascript html string

edited May 25 '15 at 3:54
Gideon
810 1 15 43

asked May 4 '09 at 22:39
Bryan
6,981 22 86 116

34 Answers

1 2 next

688

If you're running in a browser, then the easiest way is just to let the browser do it for you...

```
function strip(html)
{
   var tmp = document.createElement("DIV");
   tmp.innerHTML = html;
   return tmp.textContent || tmp.innerText || "";
}
```

Note: as folks have noted in the comments, this is best avoided if you don't control the source of the HTML (for example, don't run this on anything that could've come from user input). For those scenarios, you can *still* let the browser do the work for you - see Saba's answer on using the now widely-available DOMParser.

edited Jan 26 '18 at 16:15

answered May 4 '09 at 22:48
Shog9 ♦
133k 32 211 228

40 Just remember that this approach is rather inconsistent and will fail to strip certain characters in certain browsers. For example, in Prototype.js, we use this approach for performance, but work around some of the deficiencies - github.com/kangax/prototype/blob/… – kangax Sep 14 '09 at 16:08

11 Remember your whitespace will be messed about. I used to use this method, and then had problems as certain product codes contained double spaces, which ended up as single spaces after I got the innerText back from the DIV. Then the product codes did not match up later in the application. – Magnus Smith Sep 17 '09 at 15:03

11 @Magnus Smith: Yes, if whitespace is a concern - or really, if you have any need for this text that doesn't directly involve the specific HTML DOM you're working with - then you're better off using one of the other solutions given here. The primary advantages of this method are that it is 1) trivial, and 2) will reliably process tags, whitespace, entities, comments, etc. in *the same way as the browser you're running in*. That's frequently useful for web client code, but not necessarily appropriate for interacting with other systems where the rules are different. – Shog9 ♦ Sep 17 '09 at 21:05

213 Don't use this with HTML from an untrusted source. To see why, try running `strip("<img onerror='alert(\"could run arbitrary JS here\")' src=bogus>")` – Mike Samuel Sep 22 '11 at 18:06

23 If html contains images(img tags), the images will be requested by the browser. That's not good. – douyw Feb 13 '13 at 6:47

502

```
myString.replace(/<[^>]*>?/gm, '');
```

edited May 30 at 9:28
Mike Samuel
96.8k 24 177 219

answered May 4 '09 at 22:42
nickf
386k 176 592 692

18 Great that it works on non browser js (like node) as well. – Daniel Ribeiro Dec 8 '10 at 19:30

4 Doesn't work for `<img src=http://www.google.com.kh/images/srpr/nav_logo27.png onload="alert(42)"` if you're injecting via `document.write` or concatenating with a string that contains a `>` before injecting via `innerHTML`. – Mike Samuel Dec 24 '10 at 15:07

28 an easy fix is to change `/<.*?>/g` to `/<[^>]*>?/g`. If you agree, please edit your post so that broken security advice doesn't get copy/pasted by naïve users like Mr. Ribeiro. – Mike Samuel Dec 27 '10 at 3:16

57 @MikeSamuel Did we decide on this answer yet? Naive user here ready to copy-paste. – Ziggy May 7 '13 at 18:32

14 @AntonioMax, I've answered this question ad nauseam, but to the substance of your question, because **security critical code shouldn't be copied & pasted.** You should download a library, and keep it up-to-date and patched so that you're secure against recently discovered vulnerabilities and to changes in browsers. – Mike Samuel Nov 27 '13 at 16:04

Simplest way:

233

```
jQuery(html).text();
```

That retrieves all the text from a string of html.

edited Aug 24 '12 at 18:18
Community ♦
1 1

answered Dec 26 '11 at 1:26
Mark
2,427 1 9 2

108 We always use jQuery for projects since invariably our projects have a lot of Javascript. Therefore we didn't add bulk, we took advantage of existing API code... – Mark Mar 14 '12 at 16:31

29 You use it, but the OP might not. the question was about Javascript NOT JQuery. – Dementic Mar 14 '12 at 16:55

97 It's still a useful answer for people who need to do the same thing as the OP (like me) and don't mind using jQuery (like me), not to mention, it could have been useful to the OP if they were considering using jQuery. The point of the site is to share knowledge. Keep in mind that the chilling effect you might have by chastising useful answers without good reason. – acjay Nov 29 '12 at 1:32

24 @Dementic shockingly, I find the threads with multiple answers to be the most useful, because often a secondary answer meets my exact needs, while the primary answer meets the general case. – Eric Goldberg Dec 14 '12 at 19:11

32 That will not work if you some part of string is not wrapped in html tag. e.g. "<b>Error:</b> Please enter a valid email" will return only "Error:" – Aamir Afridi Feb 5 '13 at 11:10

81

I would like to share an edited version of the **Shog9's approved answer**.

As **Mike Samuel** pointed with a comment, that function can execute inline javascript codes.
But **Shog9** is right when saying "let the browser do it for you..."

so.. here my edited version, using DOMParser:

```
function strip(html){
   var doc = new DOMParser().parseFromString(html, 'text/html');
   return doc.body.textContent || "";
}
```

here the code to test the inline javascript:

```
strip("<img onerror='alert(\"could run arbitrary JS here\")' src=bogus>")
```

Also, it does not request resources on parse (like images)

```
strip("Just text <img src='https://assets.rbl.ms/4155638/980x.jpg'>")
```

edited Dec 6 '17 at 10:15

answered Nov 6 '17 at 15:46
Sabaz
1,901 11 23

1 It's worth to add that this solution work only in browser. – kris_IV Feb 9 '18 at 8:08

1 This is not strip tags, but more like PHP htmlspecialchars(). Still useful for me. – Daantje Sep 14 '18 at 19:38

Note that this also removes whitespace from the beginning of the text. – Raine Revere Apr 11 at 15:48

Also to note, this does work in Web Workers – Chris Seufert yesterday

52

As an extension to the jQuery method, if your string might not contian HTML (eg if you are trying to remove HTML from a form field)

```
jQuery(html).text();
```

will return an empty string if there is no html

Use:

```
jQuery('<p>' + html + '</p>').text();
```

instead.

Update: As has been pointed out in the comments, in some circumstances this solution will execute javascript contained within `html` if the value of `html` could be influenced by an attacker, use a different solution.

edited Jun 18 '17 at 13:06

answered Jan 15 '13 at 12:20
user999305
835 8 14

12 Or `$("<p>").html(html).text();` – Dimitar Dimitrov Aug 13 '14 at 15:49

3 This still executes probably dangerous code `jQuery('<span>Text :) <img src="a" onerror="alert(1)"></span>').text()` – Simon Sep 7 '16 at 10:42

try jQuery("aa&#X003c;script>alert(1)&#X003c;/script>a").text(); – Grzegorz Kaczan Jul 31 '17 at 7:17

Converting HTML for Plain Text emailing keeping hyperlinks (a href) intact

35

The above function posted by hypoxide works fine, but I was after something that would basically convert HTML created in a Web RichText editor (for example FCKEditor) and clear out all HTML but leave all the Links due the fact that I wanted both the HTML and the plain text version to aid creating the correct parts to an STMP email (both HTML and plain text).

After a long time of searching Google myself and my collegues came up with this using the regex engine in Javascript:

```
str='this string has <i>html</i> code i want to <b>remove</b><br>Link Number 1 -><a
href="http://www.bbc.co.uk">BBC</a> Link Number 1<br><p>Now back to normal text and
stuff</p>
';
str=str.replace(/<br>/gi, "\n");
str=str.replace(/<p.*>/gi, "\n");
str=str.replace(/<a.*href="(.*?)".*>(.*?)<\/a>/gi, " $2 (Link->$1) ");
str=str.replace(/<(?:.|\s)*?>/g, "");
```

the `str` variable starts out like this:

```
this string has <i>html</i> code i want to <b>remove</b><br>Link Number 1 -><a
href="http://www.bbc.co.uk">BBC</a> Link Number 1<br><p>Now back to normal text and
stuff</p>
```

and then after the code has run it looks like this:-

```
this string has html code i want to remove
Link Number 1 -> BBC (Link->http://www.bbc.co.uk)  Link Number 1


Now back to normal text and stuff
```

As you can see the all the HTML has been removed and the Link have been persevered with the hyperlinked text is still intact. Also I have replaced the `<p>` and `<br>` tags with `\n` (newline char) so that some sort of visual formatting has been retained.

To change the link format (eg. `BBC (Link->http://www.bbc.co.uk)`) just edit the `$2 (Link->$1)` , where `$1` is the href URL/URI and the `$2` is the hyperlinked text. With the links directly in body of the plain text most SMTP Mail Clients convert these so the user has the ability to click on them.

Hope you find this useful.

edited Jun 18 '15 at 14:21
Victor
1,070 12 16

answered Aug 6 '09 at 8:30
Jibberboy2000
394 3 6

It doesn't handle "&nbsp;" – Rose Nettoyeur Nov 30 '18 at 12:38

31

An improvement to the accepted answer.

```
function strip(html)
{
   var tmp = document.implementation.createHTMLDocument("New").body;
   tmp.innerHTML = html;
   return tmp.textContent || tmp.innerText || "";
}
```

This way something running like this will do no harm:

```
strip("<img onerror='alert(\"could run arbitrary JS here\")' src=bogus>")
```

Firefox, Chromium and Explorer 9+ are safe. Opera Presto is still vulnerable. Also images mentioned in the strings are not downloaded in Chromium and Firefox saving http requests.

edited Sep 19 '18 at 15:26

answered Jul 31 '13 at 20:14
Janghou
841 12 19

This is some of the way there, but isn't safe from `<script><script>alert();` – Arth Apr 21 '16 at 15:53

1 That doesn't run any scripts here in Chromium/Opera/Firefox on Linux, so why isn't it safe? – Janghou Apr 22 '16 at 10:37

My apologies, I must have miss-tested, I probably forgot to click run again on the jsFiddle. – Arth Apr 22 '16 at 10:59

The "New" argument is superfluous, I think? – Jon Schneider Dec 14 '16 at 21:43

According to the specs it's optional nowadays, but it wasn't always. – Janghou Dec 15 '16 at 12:38

18

This should do the work on any Javascript environment (NodeJS included). `text.replace(/<[^>]+>/g, '');`

answered Jan 20 '17 at 5:49
Karl.S
1,045 1 10 25

good, but leaves in things like inline stylesheets/css. – pstanton Feb 6 '18 at 5:48

@pstanton could you give a working example of your statement ? – Karl.S Feb 6 '18 at 22:42

`<html><style..>* {font-family:comic-sans;}</style>Some Text</html>` – pstanton Feb 7 '18 at 0:19

15

I altered Jibberboy2000's answer to include several `<BR />` tag formats, remove everything inside `<SCRIPT>` and `<STYLE>` tags, format the resulting HTML by removing multiple line breaks and spaces and convert some HTML-encoded code into normal. After some testing it appears that you can convert most of full web pages into simple text where page title and content are retained.

In the simple example,

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<!--comment-->

<head>

<title>This is my title</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<style>

    body {margin-top: 15px;}
    a { color: #D80C1F; font-weight:bold; text-decoration:none; }
```

```
</style>
</head>

<body>
    <center>
        This string has <i>html</i> code i want to <b>remove</b><br>
        In this line <a href="http://www.bbc.co.uk">BBC</a> with link is mentioned.
<br/>Now back to &quot;normal text&quot; and stuff using &lt;html encoding&gt;
    </center>
</body>
</html>
```

becomes

> This is my title
>
> This string has html code i want to remove
>
> In this line BBC (http://www.bbc.co.uk) with link is mentioned.
>
> Now back to "normal text" and stuff using

The JavaScript function and test page look this:

```
function convertHtmlToText() {
    var inputText = document.getElementById("input").value;
    var returnText = "" + inputText;

    //-- remove BR tags and replace them with line break
    returnText=returnText.replace(/<br>/gi, "\n");
    returnText=returnText.replace(/<br\s\/>/gi, "\n");
    returnText=returnText.replace(/<br\/>/gi, "\n");

    //-- remove P and A tags but preserve what's inside of them
    returnText=returnText.replace(/<p.*>/gi, "\n");
    returnText=returnText.replace(/<a.*href="(.*?)".*>(.*?)<\/a>/gi, " $2 ($1)");

    //-- remove all inside SCRIPT and STYLE tags
    returnText=returnText.replace(/<script.*>[\w\W]{1,}(.*?)[\w\W]{1,}<\/script>/gi,
"");
    returnText=returnText.replace(/<style.*>[\w\W]{1,}(.*?)[\w\W]{1,}<\/style>/gi, "");
    //-- remove all else
    returnText=returnText.replace(/<(?:.|\s)*?>/g, "");
    //-- get rid of more than 2 multiple line breaks:
    returnText=returnText.replace(/(?:(?:\r\n|\r|\n)\s*){2,}/gim, "\n\n");

    //-- get rid of more than 2 spaces:
    returnText = returnText.replace(/ +(?= )/g,'');

    //-- get rid of html-encoded characters:
    returnText=returnText.replace(/&nbsp;/gi," ");
    returnText=returnText.replace(/&amp;/gi,"&");
    returnText=returnText.replace(/&quot;/gi,'"');
    returnText=returnText.replace(/&lt;/gi,'<');
    returnText=returnText.replace(/&gt;/gi,'>');

    //-- return
    document.getElementById("output").value = returnText;
}
```

It was used with this HTML:

```
<textarea id="input" style="width: 400px; height: 300px;"></textarea><br />
<button onclick="convertHtmlToText()">CONVERT</button><br />
<textarea id="output" style="width: 400px; height: 300px;"></textarea><br />
```

edited May 23 '17 at 11:54
Community ♦
1 1

answered Jan 10 '12 at 12:59
Elendurwen
507 6 10

I like this solution because it has treatment of html special characters... but still not nearly enough of them... the best answer for me would deal with all of them. (which is probably what jquery does). – Daniel Gerson Oct 17 '12 at 13:17

2 I think `/<p.*>/gi` should be `/<p.*?>/gi` . – cbron May 5 '15 at 0:00

Note that to remove all `<br>` tags you could use a good regular expression instead: `/<br\s*\/?>/` that way you have just one replace instead of 3. Also it seems to me that except for the decoding of entities you can have a single regex, something like this: `/<[a-z].*?\/?>/` . – Alexis Wilke Jan 14 '16 at 7:11

Nice script. But what about table content? Any idea how can it be displayed – Hristo Enev Aug 16 '17 at 12:14

```
var text = html.replace(/<\/?("[^"]*"|'[^']*'|[^>])*(>|$)/g, "");
```

9

This is a regex version, which is more resilient to malformed HTML, like:

Unclosed tags

```
Some text <img
```

"<", ">" inside tag attributes

```
Some text <img alt="x > y">
```

Newlines

```
Some <a
href="http://google.com">
```

The code

```
var html = '<br>This <img alt="a>b" \r\n src="a_b.gif" />is > \nmy<>< > <a>"text"</a'
var text = html.replace(/<\/?("[^"]*"|'[^']*'|[^>])*(>|$)/g, "");
```

answered Jul 6 '18 at 10:39
hegemon
4,883 25 28

stackoverflow.com/a/1732454/6311260 – Berry M. Mar 25 at 10:04

7

Another, admittedly less elegant solution than nickf's or Shog9's, would be to recursively walk the DOM starting at the <body> tag and append each text node.

```
var bodyContent = document.getElementsByTagName('body')[0];
var result = appendTextNodes(bodyContent);

function appendTextNodes(element) {
    var text = '';

    // Loop through the childNodes of the passed in element
    for (var i = 0, len = element.childNodes.length; i < len; i++) {
        // Get a reference to the current child
        var node = element.childNodes[i];
```

```
            // Append the node's value if it's a text node
            if (node.nodeType == 3) {
                text += node.nodeValue;
            }
            // Recurse through the node's children, if there are any
            if (node.childNodes.length > 0) {
                appendTextNodes(node);
            }
        }
        // Return the final result
        return text;
    }
```

answered May 4 '09 at 23:14
Bryan
2,646 7 34 40

2 yikes. if you're going to create a DOM tree out of your string, then just use shog's way! – nickf May 4 '09 at 23:21

Yes, my solution wields a sledge-hammer where a regular hammer is more appropriate :-). And I agree that yours and Shog9's solutions are better, and basically said as much in the answer. I also failed to reflect in my response that the html is already contained in a string, rendering my answer essentially useless as regards the original question anyway. :-(– Bryan May 5 '09 at 0:08

1 To be fair, this has value - if you absolutely must preserve /all/ of the text, then this has at least a decent shot at capturing newlines, tabs, carriage returns, etc... Then again, nickf's solution should do the same, and do much faster... eh. – Shog9 ♦ May 5 '09 at 4:58

6

If you want to keep the links and the structure of the content (h1, h2, etc) then you should check out TextVersionJS You can use it with any HTML, although it was created to convert an HTML email to plain text.

The usage is very simple. For example in node.js:

```
var createTextVersion = require("textversionjs");
var yourHtml = "<h1>Your HTML</h1><ul><li>goes</li><li>here.</li></ul>";

var textVersion = createTextVersion(yourHtml);
```

Or in the browser with pure js:

```
<script src="textversion.js"></script>
<script>
```

```
  var yourHtml = "<h1>Your HTML</h1><ul><li>goes</li><li>here.</li></ul>";
  var textVersion = createTextVersion(yourHtml);
</script>
```

It also works with require.js:

```
define(["textversionjs"], function(createTextVersion) {
  var yourHtml = "<h1>Your HTML</h1><ul><li>goes</li><li>here.</li></ul>";
  var textVersion = createTextVersion(yourHtml);
});
```

answered Aug 4 '16 at 7:38
gyula.nemeth
709 9 8

4

After trying all of the answers mentioned most if not all of them had edge cases and couldn't completely support my needs.

I started exploring how php does it and came across the php.js lib which replicates the strip_tags method here: http://phpjs.org/functions/strip_tags/

answered Jun 11 '15 at 22:06
Deminetix
1,772 17 18

This is a neat function and well documented. However, it can be made faster when `allowed == ''` which I think is what the OP asked for, which is nearly what Byron answered below (Byron only got the `[^>]` wrong.) – Alexis Wilke Jan 14 '16 at 8:08

1 If you use the `allowed` param you are vulnerable to XSS: `stripTags('<p onclick="alert(1)">mytext</p>', '<p>')` returns `<p onclick="alert(1)">mytext</p>` – Chris Cinelli Feb 20 '16 at 1:26

4

```
function stripHTML(my_string){
    var charArr   = my_string.split(''),
        resultArr = [],
        htmlZone  = 0,
        quoteZone = 0;
    for( x=0; x < charArr.length; x++ ){
     switch( charArr[x] + htmlZone + quoteZone ){
       case "<00" : htmlZone  = 1;break;
       case ">10" : htmlZone  = 0;resultArr.push(' ');break;
       case '"10' : quoteZone = 1;break;
       case "'10" : quoteZone = 2;break;
       case '"11' :
       case "'12" : quoteZone = 0;break;
       default    : if(!htmlZone){ resultArr.push(charArr[x]); }
     }
    }
    return resultArr.join('');
}
```

Accounts for > inside attributes and `<img onerror="javascript">` in newly created dom elements.

usage:

```
clean_string = stripHTML("string with <html> in it")
```

demo:

https://jsfiddle.net/gaby_de_wilde/pqayphzd/

demo of top answer doing the terrible things:

https://jsfiddle.net/gaby_de_wilde/6f0jymL6/1/

edited Mar 27 '16 at 7:29

community wiki
3 revs, 2 users 91%
user40521

You'll need to handle escaped quotes inside an attribute value too (e.g. `string with <a malicious="attribute \">this text should be removed, but is not">example</a>`). – Logan Pickup Oct 25 '17 at 22:00

4

A lot of people have answered this already, but I thought it might be useful to share the function I wrote that strips HTML tags from a string but allows you to include an array of tags that you do not want stripped. It's pretty short and has been working nicely for me.

```
function removeTags(string, array){
  return array ? string.split("<").filter(function(val){ return f(array, val);
}).map(function(val){ return f(array, val); }).join("") : string.split("
<").map(function(d){ return d.split(">").pop(); }).join("");
  function f(array, value){
    return array.map(function(d){ return value.includes(d + ">"); }).indexOf(true) != -1
? "<" + value : value.split(">")[1];
  }
}

var x = "<span><i>Hello</i> <b>world</b>!</span>";
console.log(removeTags(x)); // Hello world!
console.log(removeTags(x, ["span", "i"])); // <span><i>Hello</i> world!</span>
```

answered Jan 27 '17 at 6:55
Harry Stevens
689 9 14

3

I think the easiest way is to just use Regular Expressions as someone mentioned above. Although there's no reason to use a bunch of them. Try:

```
stringWithHTML = stringWithHTML.replace(/<\/?[a-z][a-z0-9]*[^<>]*>/ig, "");
```

answered Jan 10 '11 at 5:40
Byron Carasco
94 2 6

11 Don't do this if you care about security. If the user input is this: '<scr<script>ipt>alert(42);</scr</script>ipt>' then the stripped version will be this: '<script>alert(42);</script>'. So this is an XSS vulnerability. – molnarg Mar 6 '13 at 12:38

You should change the `[^<>]` with `[^>]` because a valid tag cannot include a `<` character, then the XSS vulnerability disappears. – Alexis Wilke Jan 14 '16 at 8:00

3

I made some modifications to original Jibberboy2000 script Hope it'll be usefull for someone

```
str = '**ANY HTML CONTENT HERE**';

str=str.replace(/<\s*br\/*>/gi, "\n");
str=str.replace(/<\s*a.*href="(.*?)".*>(.*?)<\/a>/gi, " $2 (Link->$1) ");
str=str.replace(/<\s*\/*.+?>/ig, "\n");
```

```
str=str.replace(/ {2,}/gi, " ");
str=str.replace(/\n+\s*/gi, "\n\n");
```

answered Oct 4 '11 at 14:02

Jaxolotl
39 1

3

Here's a version which sorta addresses @MikeSamuel's security concern:

```
function strip(html)
{
   try {
       var doc = document.implementation.createDocument('http://www.w3.org/1999/xhtml',
'html', null);
       doc.documentElement.innerHTML = html;
       return doc.documentElement.textContent||doc.documentElement.innerText;
   } catch(e) {
       return "";
   }
}
```

Note, it will return an empty string if the HTML markup isn't valid XML (aka, tags must be closed and attributes must be quoted). This isn't ideal, but does avoid the issue of having the security exploit potential.

If not having valid XML markup is a requirement for you, you could try using:

```
var doc = document.implementation.createHTMLDocument("");
```

but that isn't a perfect solution either for other reasons.

edited Jul 12 '12 at 21:10

answered Jul 12 '12 at 20:38

Jeremy Johnstone
348 1 6

That will fail in many circumstances if the text comes from user input (textarea or contenteditable widget...) – Alexis Wilke Jan 14 '16 at 7:12

3

You can safely strip html tags using the iframe sandbox attribute.

The idea here is that instead of trying to regex our string, we take advantage of the browser's native parser by injecting the text into a DOM element and then querying the `textContent` / `innerText` property of that element.

The best suited element in which to inject our text is a sandboxed iframe, that way we can prevent any arbitrary code execution (Also known as XSS).

The downside of this approach is that it only works in browsers.

Here's what I came up with (Not battle-tested):

```
const stripHtmlTags = (() => {
  const sandbox = document.createElement("iframe");
  sandbox.sandbox = "allow-same-origin"; // <--- This is the key
  sandbox.style.setProperty("display", "none", "important");

  // Inject the sanbox in the current document
  document.body.appendChild(sandbox);

  // Get the sandbox's context
  const sanboxContext = sandbox.contentWindow.document;

  return (untrustedString) => {
    if (typeof untrustedString !== "string") return "";

    // Write the untrusted string in the iframe's body
    sanboxContext.open();
    sanboxContext.write(untrustedString);
    sanboxContext.close();

    // Get the string without html
    return sanboxContext.body.textContent || sanboxContext.body.innerText || "";
  };
})();
```

Usage (demo):

```
console.log(stripHtmlTags(`<img onerror='alert("could run arbitrary JS here")'
src='bogus'>XSS injection :)`));
console.log(stripHtmlTags(`<script>alert("awdawd");</` + `script>Script tag injection
:)`));
console.log(stripHtmlTags(`<strong>I am bold text</strong>`));
console.log(stripHtmlTags(`<html>I'm a HTML tag</html>`));
console.log(stripHtmlTags(`<body>I'm a body tag</body>`));
console.log(stripHtmlTags(`<head>I'm a head tag</head>`));
console.log(stripHtmlTags(null));
```

edited Apr 4 '18 at 18:20

answered Apr 4 '18 at 16:48
Etienne Martin
3,202 1 21 38

Great solution for web based environments! You should probably not be using an IIFE as since ECMAScript 2015, block-scoped variables are already scoped to the block properly with the `let` and `const` operators. Also, using your solution, I got plenty references of `iframes` not used inside the document. Consider adding a `document.body.removeChild(sandbox)` in the code for future copy-pasta based readers. – Amin NAIRI 21 hours ago

2

With jQuery you can simply retrieving it by using

```
$('#elementID').text()
```

answered Sep 3 '12 at 15:03
ianaz
1,473 1 20 31

2

Below code allows you to retain some html tags while stripping all others

```
function strip_tags(input, allowed) {

  allowed = (((allowed || '') + '')
    .toLowerCase()
    .match(/<[a-z][a-z0-9]*>/g) || [])
    .join(''); // making sure the allowed arg is a string containing only tags in
lowercase (<a><b><c>)

  var tags = /<\/?([a-z][a-z0-9]*)\b[^>]*>/gi,
      commentsAndPhpTags = /<!--[\s\S]*?-->|<\?(?:php)?[\s\S]*?\?>/gi;

  return input.replace(commentsAndPhpTags, '')
      .replace(tags, function($0, $1) {
          return allowed.indexOf('<' + $1.toLowerCase() + '>') > -1 ? $0 : '';
      });
}
```

answered Jul 14 '15 at 12:56

aWebDeveloper
20.5k 29 124 203

1 You should quote the source (`phpjs`). If you use the `allowed` param you are vulnerable to XSS: `stripTags('<p onclick="alert(1)">mytext</p>', '<p>')` returns `<p onclick="alert(1)">mytext</p>` – Chris Cinelli Feb 20 '16 at 1:25

2

It is also possible to use the fantastic htmlparser2 pure JS HTML parser. Here is a working demo:

```
var htmlparser = require('htmlparser2');

var body = '<p><div>This is </div>a <span>simple </span> <img src="test"></img>example.
</p>';

var result = [];

var parser = new htmlparser.Parser({
    ontext: function(text){
        result.push(text);
    }
}, {decodeEntities: true});

parser.write(body);
parser.end();

result.join('');
```

The output will be `This is a simple example.`

See it in action here: https://tonicdev.com/jfahrenkrug/extract-text-from-html

This works in both node and the browser if you pack you web application using a tool like webpack.

answered Dec 29 '15 at 19:11

Johannes Fahrenkrug
32.7k 16 102 144

2

I just needed to strip out the `<a>` tags and replace them with the text of the link.

This seems to work great.

```
htmlContent= htmlContent.replace(/<a.*href="(.*?)">/g, '');
htmlContent= htmlContent.replace(/<\/a>/g, '');
```

edited Jan 6 '16 at 18:57

answered Aug 19 '13 at 16:12
FrigginGlorious
89 2 6

This only applies for a tags and needs tweaking for being a wide function. – erm3nda Jan 6 '16 at 11:03

Yeah, plus an anchor tag could have many other attributes such as the `title="..."` . – Alexis Wilke Jan 14 '16 at 7:58

1

I have created a working regular expression myself:

```
str=str.replace(/(<\?[a-z]*(\s[^>]*)?\?(>|$)|<!\[[a-z]*\[|\]\]>|<!DOCTYPE[^>]*?(>|$)|<!-
-[\s\S]*?(-->|$)|<[a-z?!\/]([a-z0-9_:.])*(\s[^>]*)?(>|$))/gi, '');
```

answered Nov 9 '12 at 16:06
MarekJ47
95 3

1

simple 2 line jquery to strip the html.

```
 var content = "<p>checking the html source&nbsp;</p><p>&nbsp;
  </p><p>with&nbsp;</p><p>all</p><p>the html&nbsp;</p><p>content</p>";

 var text = $(content).text();//It gets you the plain text
 console.log(text);//check the data in your console

 cj("#text_area_id").val(text);//set your content to text area using text_area_id
```

answered Jul 5 '13 at 9:18

Developer
2,504 4 28 39

1

The accepted answer works fine mostly, however in IE if the `html` string is `null` you get the `"null"` (instead of ''). Fixed:

```
function strip(html)
{
   if (html == null) return "";
   var tmp = document.createElement("DIV");
   tmp.innerHTML = html;
   return tmp.textContent || tmp.innerText || "";
}
```

answered May 27 '16 at 0:12
basarat
151k 28 286 387

1

Using Jquery:

```
function stripTags() {
    return $('<p></p>').html(textToEscape).text()
}
```

answered Dec 9 '16 at 8:41
math2001
2,080 14 29

1

`input` element support only one line text:

> The text state represents a one line plain text edit control for the element's value.

```
function stripHtml(str) {
  var tmp = document.createElement('input');
  tmp.value = str;
  return tmp.value;
}
```

Update: this works as expected

```
function stripHtml(str) {
  // Remove some tags
  str = str.replace(/<[^>]+>/gim, '');

  // Remove BB code
  str = str.replace(/\[(\w+)[^\]]*](.*?)\[\/\1]/g, '$2 ');

  // Remove html and line breaks
  const div = document.createElement('div');
  div.innerHTML = str;

  const input = document.createElement('input');
  input.value = div.textContent || div.innerText || '';

  return input.value;
}
```

edited Oct 27 '17 at 2:13

answered Jun 14 '17 at 14:32

Mike Datsko

115 10

Doesn't work, please always mention the browser you are using when posting an answer. This is inaccurate and won't work in Chrome 61. Tags are just rendered as a string. – vdegenne Oct 2 '17 at 13:26

0

```
(function($){
    $.html2text = function(html) {
        if($('#scratch_pad').length === 0) {
            $('<div id="lh_scratch"></div>').appendTo('body');
        }
        return $('#scratch_pad').html(html).text();
    };

})(jQuery);
```

Define this as a jquery plugin and use it like as follows:

```
$.html2text(htmlContent);
```

answered Mar 16 '12 at 6:25
Shiv Shankar
9 1

Lets say this comes from user input. It can be used to add script or macros to your page – Oluwatumbi Jul 14 '18 at 13:26

0

For escape characters also this will work using pattern matching:

```
myString.replace(/((&lt)|(<)(?:.|\n)*?(&gt)|(>))/gm, '');
```

edited Nov 16 '16 at 6:00

answered Nov 8 '16 at 10:44
Abhishek Dhanraj Shahdeo
933 6 26

1 2 next

protected by Samuel Liew ♦ Apr 5 '18 at 3:32

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?