

Podcast: We chat with Major League Hacking about all-nighters, cup stacking, and therapy dogs. [Listen now](#).

# How do I enumerate all of the html id's in a document with javascript?

Asked 8 years, 4 months ago   Active 1 year, 7 months ago   Viewed 31k times



35



10

I would like to be able to use javascript to find every id (or name) for every object in an html document so that they can be printed at the bottom of the page.

To understand more fully what I'm trying to accomplish, let me explain. I build large forms from time to time for things such as property applications, rental listings, detailed medical website user registration forms and such. As I do it now, I build the form, assign the id's and names and decide which values are required and such. Then when I build the php form validation and database insert portion for the form, I've been manually going through the html and pulling out all of the id's to reference from the \$\_post array for the input validation and database insert. This has been very time consuming and a real pain, often laced with typing errors.

The form I'm working on currently is just too big, and I'd much rather write a javascript function that I can run on my local copy of the page to list all of the id's so that I don't have to copy and paste them one by one, or write them down. I could then also use the javascript loop to event print out the php code around the id names so that I'd only have to copy the list and lightly edit out the id's I didn't need... I hope you guys get the idea.

Any suggestions on how I can drop all of the id's into an array, or if there is already an array I can access and loop through (I couldn't find anything on google). Also, any suggestions for how to speed up the process of producing large forms with a work flow that generates the php or makes it quicker than my current method would be greatly appreciated!

javascript

html

validation

edited Sep 6 '15 at 20:12



[Willi Mentzel](#)

15k 11 63 82

asked Aug 18 '11 at 22:42



[rmmoul](#)

2,595 3 18 31

## 8 Answers



On [modern browsers](#) you can do this via

50

```
document.querySelectorAll('*[id]')
```

should do the job.

If you need all descendants of `myElement` with IDs, then do

```
myElement.querySelectorAll('*[id]')
```

If you want to be really careful to exclude `<span id="">`, then maybe

```
document.querySelectorAll('*[id]:not([id=""])')
```

If compatibility with older browsers is required

```
var allElements = document.getElementsByTagName("*");
var allIds = [];
for (var i = 0, n = allElements.length; i < n; ++i) {
    var el = allElements[i];
    if (el.id) { allIds.push(el.id); }
}
```

should leave you with all the IDs in `allIds`.

If you find you need to just enumerate the IDs under a particular form node, then you can replace `document.getElementsByTagName` with `myFormNode.getElementsByTagName`.

If you want to include both IDs and NAMEs, then put

```
else if (el.name) { allIds.push(el.name); }
```

below the `if` above.

edited Apr 8 '17 at 14:39

answered Aug 18 '11 at 22:44



Mike Samuel

101k

26

186

222

I keep searching for `*[id]` on the internet, I can't find anything, where do I read on it? why does it return all ID's and why doesn't it work on `var s = new Array; s[s.length] = document.getElementById("*[id]")` (inside a for loop) – [Shayan](#) Mar 22 at 1:33

- 1 [@Shayan](#), `getElementById` takes an ID not a CSS selector so you're passing the wrong kind of thing. [Mozdev](#) explains CSS selectors. `*` is a [universal selector](#) and `[id]` is an [attribute selector](#). – [Mike Samuel](#) Mar 22 at 20:22

If you're doing your development using a fairly modern browser, you can use `querySelectorAll()`, then use `Array.prototype.forEach` to iterate the collection.

19

```
var ids = document.querySelectorAll('[id]');

Array.prototype.forEach.call( ids, function( el, i ) {
    // "el" is your element
    console.log( el.id ); // Log the ID
});
```

If you want an Array of IDs, then use `Array.prototype.map`:

```
var arr = Array.prototype.map.call( ids, function( el, i ) {
    return el.id;
});
```

answered Aug 18 '11 at 22:51



[user113716](#)

279k 59 415 428

- 3 In 2015, this is the best answer. – [bryanbraun](#) Mar 6 '15 at 4:12

Get all tags with the wildcard:

10

```
var allElements = document.getElementsByTagName('*');
for(var i = 0; i < allElements.length; i++) {
    // ensure the element has an Id that is not empty and does exist
    // and string other than empty, '', will return true
```

```
allElements[i].id && console.log(allElements[i].id);
}
```

edited Feb 7 '14 at 15:03

answered Aug 18 '11 at 22:46



Joe

66.9k

15

112

135

The jQuery selector `$('[id]')` will get all the elements with an `id` attribute:

9

```
$('[id]').each(function () {
    do_something(this.id);
});
```

Working example here: <http://jsfiddle.net/RichieHindle/yzMjJ/2/>

answered Aug 18 '11 at 22:48



RichieHindle

217k

40

320

380

I forgot about the attribute selectors, much simpler – Russ Cam Aug 18 '11 at 22:50



well, since it is a form, im sure that you want to iterate only over the form elements and not all the tags in the document ( like href , div's etc.. )

5

```
for (var i=0; i < form.elements.length; i++) {
    var elementId = form.elements[i].id;
}
```

edited Dec 23 '13 at 11:33

answered Aug 18 '11 at 23:00



Dementic

12.9k

15

57

86

with jQuery

3

```
$('#*').map(function() {  
    return this.id || null;  
}).get().join(',');
```

this gets all the elements in the DOM, and runs a function on each to return the id (and if `undefined`, returning `null` won't return anything. This returns a jQuery object which is then converted to a JavaScript array with `get()` and this is then converted to a comma-separated string of ids.

Try it on this page and you get

"notify-container,overlay-header,custom-header,header,portalLink,topbar,hlinks,hlinks-user,hlinks-nav,hlinks-custom,hsearch,search,hlogo,hmenus,nav-questions,nav-tags,nav-users,nav-badges,nav-unanswered,nav-askquestion,content,question-header,mainbar,question,edit-tags,link-post-7115022,close-question-7115022,flag-post-7115022,comments-7115022,add-comment-7115022,comments-link-7115022,answers,answers-header,tabs,answer-7115033,link-post-7115033,flag-post-7115033,comments-7115033,add-comment-7115033,comments-link-7115033,answer-7115042,link-post-7115042,flag-post-7115042,comments-7115042,add-comment-7115042,comments-link-7115042,answer-7115043,link-post-7115043,delete-post-7115043,flag-post-7115043,post-editor-7115043,wmd-button-bar-7115043,wmd-button-row-7115043,wmd-bold-button-7115043,wmd-italic-button-7115043,wmd-spacer1-7115043,wmd-link-button-7115043,wmd-quote-button-7115043,wmd-code-button-7115043,wmd-image-button-7115043,wmd-spacer2-7115043,wmd-olist-button-7115043,wmd-ulist-button-7115043,wmd-heading-button-7115043,wmd-hr-button-7115043,wmd-spacer3-7115043,wmd-undo-button-7115043,wmd-redo-button-7115043,wmd-help-button-7115043,wmd-input-7115043,draft-saved-7115043,communitymode-7115043,wmd-preview-7115043,fkey,author,edit-comment-7115043,edit-comment-error-7115043,submit-button-7115043,comments-7115043,add-comment-7115043,comments-link-7115043,post-form,post-editor,wmd-button-bar,wmd-input,draft-saved,communitymode,wmd-preview,fkey,author,submit-button,show-editor-button,sidebar,qinfo,adzerk2,newsletter-ad,newsletter-ad-header,newsletter-signup-container,newsletter-signup,newsletter-preview-container,newsletter-preview,h-related,feed-link,feed-link-text,pretty-lang,footer,footer-menu,footer-sites,footer-flair,svnrev,copyright"

answered Aug 18 '11 at 22:46



**Russ Cam**

110k

24

174

236

A simple ES6 (es2015) solution based on answer of [user113716](#)

0

```
const elementsById = document.querySelectorAll('[id]');
const elementsByIdArr = Array.prototype.map.call(elementsById, el => el.id);
```

[Show code snippet](#)

answered May 18 '18 at 9:09



[Theuns Coetzee](#)

13 4

First of all, I would highly recommend jQuery. It has simplified my JavaScript development soooo much. (See RichieHindle's answer)

-1

Second, I know that a lot of browsers keep a list of IDs for direct (fast) access, but I don't know of a way to access them. It would probably be browser-specific anyways, so that's probably not the best route.

Finally, the code:

```
var elementList = document.getElementsByTagName("*");
var idList = [];
for (var i in elementList) {
  if (elementList[i].id != "") {
    idList.push(elementList[i].id);
  }
}
// Do something with your array of ids
```

edited Aug 18 '11 at 23:17



[RichieHindle](#)

217k 40 320 380

answered Aug 18 '11 at 22:52



[RustyTheBoyRobot](#)

5,359 3 28 51

@patrick Haha, oops. It's been too long since I've used vanilla JavaScript. Edited – [RustyTheBoyRobot](#) Aug 18 '11 at 22:58

- 1 It also will not work because there are at least three enumerable properties of the [NodeList](#) returned by [getElementsByTagName](#) that are not nodes (or elements), so the list of ids will contain three empty members that, when passed to [getElementById](#), will return null. The OP may not be expecting that. – [RobG](#) Aug 18 '11 at 23:34

@RobG: Interesting; I didn't know that. – [RustyTheBoyRobot](#) Aug 18 '11 at 23:39

@RobG: So, is the check `if (elementList[i].id != "")` not enough? Can text nodes or comment nodes or nodes other than elements have an ID specified? – [RustyTheBoyRobot](#) Aug 18 '11 at 23:48

No, because NodeList properties that don't have an id property (e.g. `NodeList.item.id`) will return `undefined`, and `undefined != ''`. But *undefined* is a valid id, so you can't simply filter it out. You could try a *hasOwnProperty* test, but there's no guarantee that a host object will follow ECMA-262 for that.

– RobG Aug 19 '11 at 0:33 

