How to check empty/undefined/null string in JavaScript?

Asked 10 years, 11 months ago Active 19 days ago Viewed 2.6m times

2574

440

I saw this thread, but I didn't see a JavaScript specific example. Is there a simple `string.Empty` available in JavaScript, or is it just a case of checking for `""` ?

javascript null is-empty

edited Dec 4 '18 at 9:53
T.Todua
34.5k 12 151 148

asked Sep 30 '08 at 17:17
casademora
31.3k 13 62 77

2 just FYI, i think the most useful APIs for the String class are at Mozilla and javascript kit. [elated.com](elated.com/articles/working-with-strings) has a tutorial on all of String's properties, methods,... Please note: the Mozilla link has been updated to developer.mozilla.org/en/JavaScript/Reference/Global_Objects/… – Gene T Oct 2 '08 at 15:16

Check this one : stackoverflow.com/a/36491147/7026966 – Himanshu Shekhar May 13 at 10:58

41 Answers

1 2 next

3235

✓

If you just want to check whether there's any value, you can do

```
if (strValue) {
    //do something
}
```

If you need to check specifically for an empty string over null, I would think checking against `""` is your best bet, using the `===` operator (so that you know that it is, in fact, a string you're comparing against).

```
}
```

edited Jun 2 '18 at 0:29
guest
177 1 7

answered Sep 30 '08 at 17:20
bdukes
120k 19 130 167

139 Testing the length property may actually be faster than testing the string against "", because the interpreter won't have to create a String object from the string literal. – Vincent Robert Oct 1 '08 at 20:07

17 @Vincent doing some naïve profiling in Chrome developer tools, testing `=== ''` vs `.length` didn't show any discernible improvement (and using `.length` only works if you can assume that you have a string) – bdukes Sep 27 '10 at 13:19

22 @bdukes when you start to care about that kind of micro-optimizations, I don't think Chrome is the browser where you are having most of your performance problems... – Vincent Robert Sep 27 '10 at 16:18

12 Just to note, if your definition of "empty string" includes whitespace, then this solution is not appropriate. A string of 1 or more spaces returns true above. If you are using JQuery you can simply use this: if ($.trim(ref).length === 0) - as per this answer to a similar question: stackoverflow.com/questions/2031085/… – CodeClimber Oct 3 '11 at 15:02

97 As expected `.length > 0` is actually **much faster** than comparing to a string literal! Check out this jsPerf – Chad Nov 28 '11 at 21:54

1039

For checking if a string is empty, null or undefined I use:

```
function isEmpty(str) {
    return (!str || 0 === str.length);
}
```

For checking if a string is blank, null or undefined I use:

```
function isBlank(str) {
    return (!str || /^\s*$/.test(str));
}
```

For checking if a string is blank or contains only white-space:

```
String.prototype.isEmpty = function() {
```

edited May 13 '14 at 14:00

Scott Harwell
6,960 2 23 40

answered Jul 16 '10 at 1:45

Jano González
10.8k 1 12 9

11 why 0 === str.length instead of str.length === 0 ? – Vincent Sep 23 '13 at 8:11

34 @Vincent Conditions are often written like this `if (variable == constant value)` and if you forget an '=' then you're assigning the constant value to the variable instead of testing. The code will still work as you can assign variable in a if. So a safer way to write this condition is to reverse the constant value and the variable. This way when you test your code you'll see an error (Invalid lef-hand side in assignment). You can also use something like JSHint to disallow assignment in conditions and be warned when you write one. – florian Sep 23 '13 at 9:58

2 shame that `/^\s*$/.test(str)` is not really readable - maybe removing spaces using simpler code or regex would be better? see stackoverflow.com/questions/6623231/… and also stackoverflow.com/questions/10800355/… – Adrien Be Jul 15 '14 at 11:33

5 /^\s*$/.test(str) can be replaced with str.trim().length === 0 – Schadenfreude Jun 19 '15 at 12:35

13 @Vincent this is also called "Yoda Conditions", like `if blue is the sky` . See dodgycoder.net/2011/11/yoda-conditions-pokemon-exception.html – AZ. Jan 26 '16 at 0:12

267

All the above are good but this will be even better. use `!!` (*not not*) operator.

```
if(!!str){
some code here;
}
```

or use type casting:

```
if(Boolean(str)){
    codes here;
}
```

Both do the same function, type cast the variable to boolean, where `str` is a variable.
Returns `false` for `null,undefined,0,000,"",false` .
Returns `true` for string "0" and whitespace " ".

edited Apr 3 '12 at 9:11

answered Mar 30 '11 at 13:34

47 Why is this "even better"? – Mene Apr 3 '11 at 13:04

24 Is there any difference between the behavior of `if(str)` and `if(!!str)` ? – Peter Olson Dec 19 '14 at 18:28

4 @PeterOlson if you are trying to save a variable as a boolean that checks multiple strings for content then you would want to do this.. aka `var any = (!!str1 && !!str2 && !!str3)` handling if there is a number in there as well – John Ruddell Mar 10 '15 at 23:00

21 This is the solution I always use. `!!str.trim()` to make sure the string is not made of whitespaces only. – Dario Oddenino Feb 11 '16 at 10:56

3 Not not looks like a hack, `Boolean(str)` is a lot more readable and less "wtfish". – shinzou Oct 17 '16 at 21:28

95

If you need to make sure that the string is not just a bunch of empty spaces (I'm assuming this is for form validation) you need to do a replace on the spaces.

```
if(str.replace(/\s/g,"") == ""){
}
```

answered Sep 30 '08 at 23:08
Sugendran
1,871 12 15

8 But does the job if what you actually want to test for is a string with non-space content. Is there a less-expensive way to test this? – flash Oct 22 '10 at 10:02

3 How about the length property? – driAn Nov 11 '10 at 13:57

24 Instead of removing all the spaces, why not just check if there's a non-space? Has 2 advantages that it can bail out early if there is a non-space character, and it doesn't have return a new string which you then check against. `if(str.match(/\S/g)){}` – mpen Jun 20 '11 at 4:29

31 @Mark FYI, you wouldn't need the global modifier, since the match of the first occurrence of a non-space character would mean the string is not empty: `str.match(/\S/)` – neezer Jun 27 '11 at 15:04

2 Important: This will throw errors, if `str` is `null` . – nfechner Feb 12 '12 at 7:54

The closest thing you can get to `str.Empty` (with the precondition that str is a String) is:

edited Aug 31 '18 at 16:08

Kick Buttowski
4,887 10 31 48

answered Sep 30 '08 at 19:17

Ates Goral
110k 22 119 180

3 Wouldn't that throw an exception is str is null? – Pic Mickael Sep 11 '18 at 13:50

2 @PicMickael Yes! So would `str.Empty` . – Ates Goral Sep 12 '18 at 3:25

54

I use :

```
function empty(e) {
  switch (e) {
    case "":
    case 0:
    case "0":
    case null:
    case false:
    case typeof this == "undefined":
      return true;
    default:
      return false;
  }
}

empty(null) // true
empty(0) // true
empty(7) // false
empty("") // true
empty((function() {
    return ""
  })) // false
```

edited Nov 13 '17 at 17:11

dun
39 5

answered Jul 9 '10 at 18:52

Jet
1,003 9 7

3 This solution is more language agnostic. The only JavaScript feature it relies on is typeof. So it is a good example of a solution you can use when you

1 I'd go even a bit further, and nail it with a === operator for the undefined case. Otherwise it's just simply the perfect answer. – xarlymg89 Jan 25 '18 at 10:06

The `typeof` in the `switch` did not work for me. I added a `if (typeof e == "undefined")` test and that works. Why? – Lucas Apr 17 '18 at 11:15

32

Try:

```
if (str && str.trim().length) {
    //...
}
```

edited Jan 9 '13 at 3:15
afuzzyllama
5,612 4 39 57

answered Jan 9 '13 at 2:54
Yang Dong
382 3 4

2 Is the length property necessary? – frogcoder Nov 22 '17 at 4:57

`str.trim().length` will do faster than `str.trim()` , by around 1% according to my own testing result. – devildelta Mar 6 at 6:32

31

```
var s; // undefined
var s = ""; // ""
s.length // 0
```

There's nothing representing an empty string in JavaScript. Do a check against either `length` (if you know that the var will always be a string) or against `""`

answered Sep 30 '08 at 17:42
cllpse
13.8k 33 122 167

Function:

```
function is_empty(x)
{
   return (
        (typeof x == 'undefined')
                    ||
        (x == null)
                    ||
        (x == false)  //same as: !x
                    ||
        (x.length == 0)
                    ||
        (x == "")
                    ||
        (x.replace(/\s/g,"") == "")
                    ||
        (!/[^\s]/.test(x))
                    ||
        (/^\s*$/.test(x))
  );
}
```

> *p.s. In Javascript, don't use Line-Break after `return` ;*

edited Jan 8 at 4:39
Lonely Neuron
3,134 3 18 35

answered Feb 27 '14 at 20:31
T.Todua
34.5k 12 151 148

1 Any chance you could explain what each check is doing? :) – DanV Mar 21 '14 at 12:13

2 -1 They are testing for different things. It makes no sense to put them all into one `if` statement. – Bennett McElwee Apr 1 '14 at 22:39

@Bennett pretty sure he's just doing it to show the different ways... – Robert Pounder Jul 17 '15 at 10:40

1 typeof MyVariable == 'undefined' doesn't discern between an initialized variable with an undefined value and an undeclared variable unless the variable was initially declared and initialized to null. Checking the length property causes the string primitive to be wrapped in a string object. – Scott Marcus Jan 28 '16 at 21:57

You can use [lodash](#) : _.isEmpty(value).

But it always returns `true` for `Number` type of Javascript Primitive Data Types like `_.isEmpty(10)` or `_.isEmpty(Number.MAX_VALUE)` both returns `true` .

edited Mar 8 '18 at 12:55

answered Aug 17 '17 at 6:51

Moshii
693 1 11 28

26

I would not worry too much about the most **efficient** method. Use what is most clear to your intention. For me that's usually `strVar == ""` .

EDIT: per comment from Constantin, if strVar could some how end up containing an integer 0 value, then that would indeed be one of those intention-clarifying situations.

edited May 23 '17 at 11:47

Community ♦
1 1

answered Sep 30 '08 at 17:20

Chris Noe
19.4k 22 64 89

11 Bad idea. You'll get true if strVar is accidentally assigned 0. – Constantin Sep 30 '08 at 19:21

5 I agree that making your intention clear is more important than any micro-optimizations other methods might yield, but using the strict comparison operator `===` would be better. It only returns true if `strVar` is an empty string. – Useless Code Jun 13 '15 at 17:52

The check fails if undefined. So if(str) works better – Valentin Heinitz Dec 27 '15 at 13:57

3 @ValentinHeinitz if str were assigned a falsey value of 0 or "0", if(str) would falsely report true. The best approach is if(str === ""). It's simple and it will never fail. – Scott Marcus Jan 28 '16 at 22:02

Which equals operator (== vs ===) should be used in JavaScript comparisons? – nobar Jan 18 '18 at 21:19

19

you could also go with regexps:

```
if((/^\s*$/).test(str)) { }
```

Checks for strings that are either empty or filled with whitespace.

oem
191 1 2

It works, but it's also horribly expensive ops-wise. Good if you just want to check one or two things, not a large set. – Orpheus May 11 '15 at 16:38

16

1. check that `var a;` exist
2. trim out the `false spaces` in the value, then test for `emptiness`

```
if ((a)&&(a.trim()!=''))
{
  // if variable a is not empty do this
}
```

edited Sep 7 '16 at 7:58
Kruti Patel
1,154 2 20 33

answered Feb 24 '15 at 15:38
Timothy Nwanwene
612 7 17

16

A lot of answers, and a lot of different possibilities!

Without a doubt for quick and simple implementation the winner is: `if (!str.length) {...}`

However, as many other examples are available. The best functional method to go about this, I would suggest:

```
function empty(str)
{
    if (typeof str == 'undefined' || !str || str.length === 0 || str === "" ||
!/[^\s]/.test(str) || /^\s*$/.test(str) || str.replace(/\s/g,"") === "")
    {
        return true;
    }
    else
    {
        return false;
    }
```

Run code snippet   Expand snippet

A bit excessive, I know.

edited Sep 14 '18 at 8:25

answered Mar 31 '16 at 8:29
tfont
7,118 4 40 40

2 Checking for undefined would need to be moved to first in the checks, or undefined items will throw exceptions on the earlier checks. – SvdSinner Jun 16 '16 at 14:10

Completely agree! NICE CATCH. I will edit my above answer! – tfont Jun 17 '16 at 15:17

14

Also, in case you consider a whitespace filled string as "empty". You can test it with this Regex:

```
!/\S/.test(string); // Returns true if blank.
```

answered May 15 '13 at 14:55
Wab_Z
141 1 4

12

I usually use some thing like this,

```
if (!str.length) {
//do some thing
}
```

answered Aug 9 '13 at 10:10
user2086641
2,221 9 42 80

@AdrianHope-Bailie why would you test an undefined variable? – Abimael Martell Apr 1 '14 at 0:18

2 @AbimaelMartell Why not? You have a variable that either you declared or that was passed to you from some scope you have no control over such as in a response from a method or API call. You can assume it contains a value and use the check above but if it is not defined or is null you will get an error. var test = null; if(!test.length){alert("adrian is wrong");} – Adrian Hope-Bailie Apr 1 '14 at 8:28

OP was asking for "how to check for an empty string", un undefined variable is not an empty string. Anyway you could check `typeof variable != "undefined"` before checking if is empty. – Abimael Martell Apr 1 '14 at 18:56

9

I have not noticed an answer that takes into account the possibility of null characters in a string. For example, if we have a null character string:

```
var y = "\0"; // an empty string, but has a null character
(y === "") // false, testing against an empty string does not work
(y.length === 0) // false
(y) // true, this is also not expected
(y.match(/^[\s]*$/)) // false, again not wanted
```

To test its nullness one could do something like this:

```
String.prototype.isNull = function(){
  return Boolean(this.match(/^[\0]*$/));
}
...
"\0".isNull() // true
```

It works on a null string, and on an empty string and it is accessible for all strings. In addition, it could be expanded to contain other JavaScript empty or whitespace characters (i.e. nonbreaking space, byte order mark, line/paragraph separator, etc.).

answered Jul 31 '12 at 14:05
Bikush
499 5 19

2 Interesting analysis. I don't think this would be relevant in 99.9% of cases. BUT I recently found that MySQL evaluates a column to "null" if (and only if) that column contains the null character ("\0"). Oracle on the other hand would not evaluate "\0" as being null, preferring to treat it as a string of length 1 (where that one character is the null character). This could cause confusion if not dealt with properly, because many web-developers do work with a back-end database, which might pass through different types of "null" values. It should be at the back of every developer's mind. – cartbeforehorse Oct

9

All these answers are nice.

But I cannot be sure that variable is a string, doesn't contains only spaces (this is important for me), and can contain '0' (string).

My version:

```
function empty(str){
    return !str || !/[^\s]+/.test(str);
}

empty(null); // true
empty(0); // true
empty(7); // false
empty(""); // true
empty("0"); // false
empty("  "); // true
```

Sample on jsfiddle.

answered May 8 '13 at 17:15

Andron

4,977 3 34 44

1 Huh? If you are expecting a string, `empty(0)` and `empty(7)` should return the same value. – Bennett McElwee Apr 1 '14 at 22:44

In my particular case - `empty("0")` must return `false` (because that is a not empty string), but `empty(0)` must return `true` because it is empty :) – Andron Apr 2 '14 at 11:24

But 0 *isn't* empty! It's a number, and numbers can't be full or empty. Of course, it's your function and so must satisfy your requirements, but `empty` is a misleading name in this case. – Bennett McElwee Apr 2 '14 at 21:15

I think that name `empty` is good. In php docs for empty function: `Returns FALSE if var exists and has a non-empty, non-zero value. Otherwise returns TRUE.` The difference between `PHP` and this function - that string `'0'` will be not identified as empty. – Andron Apr 3 '14 at 12:14

As I say, it's your function: call it what you want. But `empty` is an inaccurate and misleading name. It's interesting that PHP also has a poorly-named `empty` function, but PHP's failings don't have anything to do with JavaScript. – Bennett McElwee Apr 3 '14 at 22:57

9

```
function isEmpty(s){
    return !s.length;
}

function isBlank(s){
    return isEmpty(s.trim());
}
```

answered Oct 1 '14 at 6:55

Josef.B
639 7 12

I usually use something like:

7

```
if (str == "") {
     //Do Something
}
else {
     //Do Something Else
}
```

answered Feb 15 '10 at 2:46

jmc734
179 1 3

`==` operator is not strict. So if `str == ""`, `str` can be `null`, `undefined`, `false`, `0`, `[]`, etc.. – RousseauAlexandre Sep 11 at 7:05

I use a combination, fastest checks are first.

7

```
function isBlank(pString){
    if (!pString || pString.length == 0) {
        return true;
    }
    // checks for a non-white space character
    // against an empty string
    return !/[^\s]+/.test(pString);
}
```

answered Jun 1 '11 at 15:40

Will
4,638 2 23 46

1 Just wondering if you could explain when the length check would be necessary? Wouldn't !pString catch anything that was null/empty string? This seems to work. var test=''; if (!test) alert('empty'); – Nicholi Oct 27 '11 at 21:49

7

Ignoring whitespace strings, you could use this to check for null, empty and undefined :

```
var obj = {};
(!!obj.str) //returns false

obj.str = "";
(!!obj.str) //returns false

obj.str = null;
(!!obj.str) //returns false
```

Concise and it works for undefined properties, although it's not the most readable.

edited Oct 11 '12 at 2:56

Stephan
28.9k 32 165 265

answered Jul 19 '12 at 2:00

mricci
553 1 5 13

7

I did some research what happens if you pass a non-string and non-empty/null value to a tester function. As many knows, (0 == "") is true in javascript, but since 0 is a value and not empty or null, you may want to test for it.

The following two functions return true only for undefined, null, empty/whitespace values and false for everything else, such as numbers, boolean, objects, expressions etc.

```
    return (value == null || value === "");
}
function IsNullOrWhiteSpace(value)
{
    return (value == null || !/\S/.test(value));
}
```

More complicated examples exists, but these are simple and give consistent results. There is no need to test for undefined, since it's included in (value == null) check. You may also mimic C# behaviour by adding them to String like this:

```
String.IsNullOrEmpty = function (value) { ... }
```

You do not want to put it in Strings prototype, because if the instance of the String-class is null, it will error:

```
String.prototype.IsNullOrEmpty = function (value) { ... }
var myvar = null;
if (1 == 2) { myvar = "OK"; } // could be set
myvar.IsNullOrEmpty(); // throws error
```

I tested with the following value array. You can loop it through to test your functions if in doubt.

```
// Helper items
var MyClass = function (b) { this.a = "Hello World!"; this.b = b; };
MyClass.prototype.hello = function () { if (this.b == null) { alert(this.a); } else {
alert(this.b); } };
var z;
var arr = [
// 0: Explanation for printing, 1: actual value
    ['undefined', undefined],
    ['(var) z', z],
    ['null', null],
    ['empty', ''],
    ['space', ' '],
    ['tab', '\t'],
    ['newline', '\n'],
    ['carriage return', '\r'],
    ['"\\r\\n"', '\r\n'],
    ['"\\n\\r"', '\n\r'],
    ['" \\t \\n "', ' \t \n '],
    ['" txt \\t test \\n"', ' txt \t test \n'],
    ['"txt"', "txt"]
    ['"1"', '1'],
    ['"1.5"', '1.5'],
    ['"1,5"', '1,5'], // valid number in some locales, not in js
    ['comma', ','],
    ['dot', '.'],
    ['".5"', '.5'],
    ['0', 0],
    ['0.0', 0.0],
    ['1', 1],
    ['1.5', 1.5],
    ['NaN', NaN],
    ['/\S/', /\S/],
    ['true', true],
    ['false', false],
    ['function, returns true', function () { return true; } ],
    ['function, returns false', function () { return false; } ],
    ['function, returns null', function () { return null; } ],
    ['function, returns string', function () { return "test"; } ],
    ['function, returns undefined', function () { } ],
    ['MyClass', MyClass],
    ['new MyClass', new MyClass()],
    ['empty object', {}],
    ['non-empty object', { a: "a", match: "bogus", test: "bogus"}],
    ['object with toString: string', { a: "a", match: "bogus", test: "bogus", toString:
function () { return "test"; } }],
    ['object with toString: null', { a: "a", match: "bogus", test: "bogus", toString:
function () { return null; } }]
];
```

answered May 14 '13 at 10:18

JHM

316 3 8

If you simply stop using == and use ===, then this solves the problem if(s === ""). – Scott Marcus Jan 28 '16 at 22:05

7

to check if is exactly an empty string:

```
if(val==="")...
if(!val)...
```

answered Jun 2 '17 at 8:33

Luca C.
4,556 47 51

7

Meanwhile we can have one function that checks for all 'empties' like **null, undefined, '', ' ', {}, []**. So I just wrote this.

```
var isEmpty = function(data) {
    if(typeof(data) === 'object'){
        if(JSON.stringify(data) === '{}' || JSON.stringify(data) === '[]'){
            return true;
        }else if(!data){
            return true;
        }
        return false;
    }else if(typeof(data) === 'string'){
        if(!data.trim()){
            return true;
        }
        return false;
    }else if(typeof(data) === 'undefined'){
        return true;
    }else{
        return false;
    }
}
```

Use cases and results.

```
console.log(isEmpty()); // true
console.log(isEmpty(null)); // true
console.log(isEmpty('')); // true
console.log(isEmpty('  ')); // true
console.log(isEmpty(undefined)); // true
console.log(isEmpty({})); // true
console.log(isEmpty([])); // true
console.log(isEmpty(0)); // false
```

answered Mar 12 '18 at 8:48

Imran
3,637 2 25 35

6

There's no `isEmpty()` method, you have to check for the type and the length:

```
if (typeof test === 'string' && test.length === 0){
  ...
```

The type check is needed in order to avoid runtime errors when `test` is `undefined` or `null` .

answered Nov 4 '16 at 22:28

Agustí Sánchez
6,681 2 23 20

5

Don't assume that the variable you check is a string. Don't assume that if this var has a length, then it's a string.

The thing is: think carefully about what your app must do and can accept. Build something robust.

If your method / function should only process a non empty string then test if the argument is a non empty string and don't do some 'trick'.

As an example of something that will explode if you follow some advices here not carefully.

```
var getLastChar = function (str) {
 if (str.length > 0)
   return str.charAt(str.length - 1)
}

getLastChar('hello')
=> "o"

getLastChar([0,1,2,3])
=> TypeError: Object [object Array] has no method 'charAt'
```

```
if (myVar === '')
  ...
```

answered Jul 15 '13 at 10:55

Kev
2,148 3 19 31

5

Try this

```
str.value.length == 0
```

edited Jan 8 at 4:39

Lonely Neuron
3,134 3 18 35

answered Jul 22 '10 at 15:30

Doug
67 1 1

3 `"".value.length` will cause an error. It should be `str.length === 0` – AndFisher Dec 14 '16 at 10:57

This trow a `TypeError` If `str` is equal to `undefined` or `null` – RousseauAlexandre Sep 11 at 7:08

5

You can easily add it to native **String** object in **JavaScript** and reuse it over and over...
Something simple like below code can do the job for you if you want to check `''` empty strings:

```
String.prototype.isEmpty = String.prototype.isEmpty || function() {
  return !(!!this.length);
}
```

Otherwise if you'd like to check both `''` empty string and `'  '` with space, you can do that by just adding `trim()`, something like the code below:

```
String.prototype.isEmpty = String.prototype.isEmpty || function() {
```

and you can call it this way:

```
''.isEmpty(); //return true
'alireza'.isEmpty(); //return false
```

edited Jan 19 at 6:19

answered Oct 7 '17 at 2:33

Alireza

61.6k 15 197 133

4

```
function tell()
{
var pass = document.getElementById('pasword').value;
var plen = pass.length;

now you can check if your string is empty as like
if(plen==0)
{
         alert('empty');
}
else
{
   alert('you entered something');
}
}


<input type='text' id='pasword' />
```

this is also a generic way to check if field is empty.

answered Aug 6 '10 at 16:28

Muhammad Salman

41 1

---

1 2 next

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?