Underscore prefix for property and method names in JavaScript

Asked 8 years, 7 months ago Active 1 month ago Viewed 151k times

209

48

Is the underscore prefix in JavaScript only a convention, like for example in Python private class methods are?

From the 2.7 Python documentation:

> “Private” instance variables that cannot be accessed except from inside an object don’t exist in Python. However, there is a convention that is followed by most Python code: a name prefixed with an underscore (e.g. _spam) should be treated as a non-public part of the API (whether it is a function, a method or a data member).

Does this also apply to JavaScript?

Take for example this JavaScript code:

```
function AltTabPopup() {
    this._init();
}

AltTabPopup.prototype = {
    _init : function() {
        ...
    }
}
```

Also, underscore prefixed variables are used.

```
    ...
    this._currentApp = 0;
    this._currentWindow = -1;
    this._thumbnailTimeoutId = 0;
    this._motionTimeoutId = 0;
    ...
```

Only conventions? Or is there more behind the underscore prefix?

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

I admit my question is quite similar to this question, but it didn't make one smarter about the significance of the underscore prefix in JavaScript.

javascript scope naming-conventions

edited May 23 '17 at 12:03
Community ♦
1 1

asked Dec 19 '10 at 18:27
Kenny Meyer
4,413 4 33 57

Also see stackoverflow.com/questions/17359885/… – Jon Onstott Jun 28 '13 at 15:54

6 Answers

8

✓

Welcome to 2019!

It appears a proposal to extend class syntax to allow for `_` prefixed variable names to be public and `#` prefixed ones, private was accepted. Chrome 74 ships with this support.

answered May 19 at 8:03
Karuhanga
614 1 6 20

240

That's only a convention. The Javascript language does not give any special meaning to identifiers starting with underscore characters.

That said, it's quite a useful convention for a language that doesn't support encapsulation out of the box. Although there is no way to prevent someone from abusing your classes' implementations, at least it does clarify your intent, and documents such behavior as being *wrong* in the first place.

edited Dec 21 '10 at 23:17

answered Dec 19 '10 at 18:32
Frédéric Hamidi
215k 31 417 434

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

Serious prob. jsfiddle.net/VmFSR As you can see there, value created name is only accessible by prefixing new value, created, using `_` i'd love to know what's going on!? why it is not `this.name` instead? – Muhammad Umer Jul 26 '13 at 23:14

1 @Muhammad Umer, I'm not sure I understand your comment. `console.log(someone._name = "Jean Dupont");` works as well as `console.log(someone.name);` , and it both assigns and evaluates the underscore-prefixed member behind the property. As you can see, theres is no guaranteed encapsulation through underscores :) – Frédéric Hamidi Jul 26 '13 at 23:25

3 By default, Visual Studio try to help you respect this. The javascript IntelliSense engine show you "private" properties, from inside the object, when using the "this" variable. But, when called from the outside, it hides all underscored attributes. – foxontherock Dec 17 '15 at 19:30

1 @Karuhanga he answered this back in 2010 - of course things have changed in 10 years – Kenny Meyer Jul 15 at 14:40

95

JavaScript actually does support encapsulation, through a method that involves hiding members in closures (Crockford). That said, it's sometimes cumbersome, and the underscore convention is a pretty good convention to use for things that are sort of private, but that you don't actually *need* to hide.

edited Jun 2 at 0:07 Bob Stein 8,189 4 53 77

answered Dec 21 '10 at 23:29 Zach 6,176 2 17 24

17 Up vote for clarifying how to achieve closures, down vote for saying underscores are good convention. So I won't vote either way :) – Jason Jun 17 '11 at 3:14

3 Hiding members in closures can sometimes hinder testability. Check out this article: adequatelygood.com/2010/7/Writing-Testable-JavaScript – Zach Lysobey Feb 5 '13 at 18:46

4 @Jason - Just curious, why you consider underscore a bad convention? – Tamás Pap May 21 '13 at 14:57

4 @TamasPap - A few reasons, but just my option: 1) A crutch to force JS into a style of other languages 2) If it's accessible, it will be used. The underscore can litter up and convolute outside code. 3) Confusing to new JS programmers. – Jason Jun 5 '13 at 2:38

9 Even with a closure, it's still *technically* possible to gain access to the so called "private" variable. The _convention at least lets devs know to do so at their own risk (or something like that). – sarink Dec 5 '13 at 5:24

JSDoc 3 allows you to annotate your functions with the `@access private` (previously the `@private` tag) which is also useful for broadcasting your intent to other developers - http://usejsdoc.org/tags-access.html

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

77 1 7

594 2 8 20

9

> "Only conventions? Or is there more behind the underscore prefix?"

Apart from privacy conventions, I also wanted to help bring awareness that the underscore prefix is also used for arguments that are dependent on independent arguments, specifically in URI anchor maps. Dependent keys always point to a map.

Example (from https://github.com/mmikowski/urianchor) :

```
$.uriAnchor.setAnchor({
  page   : 'profile',
  _page  : {
    uname   : 'wendy',
    online  : 'today'
  }
});
```

The URI anchor on the browser search field is changed to:

```
\#!page=profile:uname,wendy|online,today
```

This is a convention used to drive an application state based on hash changes.

edited Apr 17 at 3:44
Kenny Meyer
4,413 4 33 57

answered Aug 26 '14 at 18:52
Sam Araiza
120 1 9

8

`import/export` is now doing the job with ES6. I still tend to prefix not exported functions with `_` if most of my functions are exported.

If you export only a class (like in angular projects), it's not needed at all.

```
export class MyOpenClass{
```

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

```
            return close();
        }

        _privateStuff() { /* _ only as a convention */}

    }

    function close(){ /*... this is really private... */ }
```

edited Oct 15 '18 at 7:08

answered Mar 24 '17 at 8:25

Nicolas Zozol
5,275 1 41 60

I don't think that import/export offers support for private class methods in any way. I mean, it does support a similar functionality on class level, but it does not offer hiding the contained methods. (i.e. all contained methods are always public) – bvdb Oct 11 '18 at 9:17

You export the class, and inner function call outside function. These functions are privates. – Nicolas Zozol Oct 15 '18 at 7:06

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.