Skip type check on unused parameters

Asked 3 years, 2 months ago Active 4 months ago Viewed 8k times

10

4

When I compile my typescript project, I'm using the `noImplicitAny` option so that I won't forget to specify the types on my variables and arguments.

However sometimes you have arguments that you don't use. For example:

```
jQuery.ajaxTransport("+*", function (options: JQueryAjaxSettings) {
  return {
    abort: function (_, callback: JQueryCallback) {
```

I am not interested in the first argument of the abort function, so I ignore it by naming it _.

Is that the proper way to do that in TypeScript? I couldn't find it in the guide. I suspect that it isn't the proper way, because I can only name one argument _.

Typescript raises the following error:

> error TS7006: Parameter '_' implicitly has an 'any' type.

I could just type `_:any` but that seems a bit overkill for an argument that I don't use.

typescript unused-variables

asked Aug 8 '16 at 17:14
Zyphrax
14k 9 57 81

1 there's no such thing as a default parameter name. if you enabled noImplicitAny you must specify the type for everything. – toskv Aug 8 '16 at 18:12

@toskv that's unfortunate, but thank you. If you add it as an answer I will accept it :) – Zyphrax Aug 8 '16 at 20:47

1 github.com/Microsoft/TypeScript/issues/14154 – glen-84 Apr 27 '18 at 19:15

2 Answers

18

✓

I was having the same problem. Using say express and routing you would often only want the res parameter.

```
router.get('/', function (req, res) { res.end('Bye.'); });
```

Your idea of using _ works here, but I've also found doing this works too.

```
function (_1, _2, _3, onlyThis) { console.log(onlyThis); }
```

This seems better, as only doing '_' I think might make using lodash/underscore a bit confusing, and it also makes it obvious it's the 4th parameter your interested in.

answered Sep 23 '16 at 12:18
Keith
10.7k 1 10 21

2 Would the numbered parameters not also have an implied `any` type? That was the problem that the OP was trying to resolve. – Michael L Perry Mar 14 '18 at 18:23

@MichaelLPerry It was to get around this problem -> `because I can only name one argument _.` The underscore trick also works for Linters too. Also at the time not sure how Typescript handled this, currently checking on an unused parameter seems a bit odd anyway, so I'd say if Typescript is still treating underscore params as unused, this warning should not appear. You would now need to do `_1:any, _2:any` .. – Keith Mar 14 '18 at 18:45

Why use numbers instead of something like `_req` ? And yes, this doesn't answer the OP's question. – glen-84 Apr 27 '18 at 18:57

There is nothing wrong in using `_req` , never said there was. In fact it's what I tend to do now, but my original answer was done Sept 2016. Not sure

9

I may be late, but I got stuck with the other solutions and this one work all the time for me:

```
function ({}={}, {}={}, {}={}, onlyThis) { console.log(onlyThis); }
```

comparison

When using the `_0` , `_1` , ... solution, I faced difficulties with scooped function like:

```
function parent(_0, _1, _2) {
  function child(_0, _1, _2) {
    // TypeScript crying about shallowed variables
  }
}
```

but with the empty object it work well:

```
function parent({}, {}, {}) {
  function child({}, {}, {}) {
    // :-)
  }
}
```

Even if the parameters are typed like:

```
function parent({}: number, {}: string, {}: any) {
  function child({}: number, {}: string, {}: any) {
    // :-) x2
  }
}
```

EDIT:

And as written here, setting a default value avoid error throwing if the given parameter is `undefined` or `null` .

```
    // :-)
  }
}
```

edited May 15 at 19:38
geisterfurz007 **2,720** 2 22 36

answered May 26 '18 at 23:56
Miaow **418** 4 14

4 This is horrible; I love it. – hraban Apr 16 at 9:22