angular2: how to copy object into another object

40

10

Please help me in order to copy object into another object using angular 2 ?

In angular i used angular.copy() to copy object to loose reference of old object. But, when i used same in angular 2 getting below error:

> Error: angular is not defined.

angular typescript

edited Apr 28 at 11:36
displayname
7,323 18 92 183

asked Sep 15 '16 at 8:42
Swetha
276 1 5 10

2 Possible duplicate of How can I use angular.copy in angular 2 – sisve Nov 30 '17 at 7:30

7 Answers

95

Solution

Angular2 developed on the ground of modern technologies like TypeScript and ES6. So you can just do `let copy = Object.assign({}, myObject)` .

Object assign - nice examples.

For nested objects : `let copy = JSON.parse(JSON.stringify(myObject))`

edited Jun 21 '17 at 11:34

answered Sep 15 '16 at 8:45

1 This would copy an object by reference. What if I do want to copy by reference, so as to avoid any changes in myObject affecting 'copy' object. – Manoj Amalraj May 8 '17 at 19:57

@PratapA.K Take a look at loadsh, if you can load another library in your system. It has functions such as _cloneDeep. – PeterS May 8 '17 at 20:19

'Object' only refers to a type, but is being used as a namespace here.ts(2702) – Omid Farvid Jun 13 at 23:36

25

```
let copy = Object.assign({}, myObject).  as mentioned above
```

but this wont work for nested objects. SO an alternative would be

```
let copy =JSON.parse(JSON.stringify(myObject))
```

answered May 26 '17 at 10:49

Ankit Raonka
1,653 4 19 38

stringify worked for my nesting needs. Thanks. – Brent Oct 31 '17 at 18:37

worked for me when nested objects are there, Thank you – Swift Nov 23 '18 at 10:55

5

You can do in this in Angular with ECMAScript6 by using the spread operator:

```
let copy = {...myObject};
```

answered Aug 22 '18 at 15:13

Kabb5
2,747 2 25 46

4

As suggested before, the clean way of deep copying objects having nested objects inside is by using lodash's cloneDeep method.

For Angular, you can do it like this:

Install lodash with `yarn add lodash` or `npm install lodash`.

In your component, import `cloneDeep` and use it:

```
import * as cloneDeep from 'lodash/cloneDeep';
...
clonedObject = cloneDeep(originalObject);
```

It's only 18kb added to your build, well worth for the benefits.

I've also written an article here, if you need more insight on why using lodash's cloneDeep.

answered Jan 15 '18 at 15:42

BogdanC
1,657 2 16 15

2

```
let course = {
  name: 'Angular',
};

let newCourse= Object.assign({}, course);

newCourse.name= 'React';

console.log(course.name); // writes Angular
console.log(newCourse.name); // writes React
```

For Nested Object we can use of 3rd party libraries, for deep copying objects. In case of lodash, use _.cloneDeep()

```
let newCourse=  .cloneDeep(course);
```

0

Loadsh is the universal standard library for coping any object deepcopy. It's a recursive algorithm. It's check everything and does copy for the given object. Writing this kind of algorithm will take longer time. It's better to leverage the same.

answered Nov 20 '18 at 18:30

Surajit Das
1 3

0

Try this.

Copy an Array :

```
const myCopiedArray  = Object.assign([], myArray);
```

Copy an object :

```
const myCopiedObject = Object.assign({}, myObject);
```

answered Jun 30 at 10:04

Saif
161 1 8