Getting the enum key with the value string (reverse mapping) in TypeScript

Asked 8 months ago Active 2 months ago Viewed 3k times

4

I have an enum:

```
export enum ApiMessages {
    logged_ok = 'Logged OK',
    register_ok = 'Register OK'
}
```

I have a function with the enum as a parameter:

```
export function responseOK(message: ApiMessages, result ?: any): ApiResponse {
    return {
        "status": "ok",
        "code": 200,
        "messageId": ApiMessages[message], <-- KO TS7015
        "message": message,
        "result": result
    };
}
```

I am calling the function like that:

```
responseOK(ApiMessages.logged_ok, {user: userRes})
```

I am trying to return the enum key and the enum string value to the response but I get the TS error:

> TS7015: Element implicitly has an 'any' type because index expression is not of type 'number'.

I have strict TypeScript config. Adding suppressImplicitAnyIndexErrors is not an option.

TypeScript version: 2.9.2

edited Jul 11 at 17:58

Erik Philips
43.3k 6 100 132

asked Jan 21 at 20:31

miqrc
352 4 13

1 Hi. In your example message is the value of the enum and not the key. So message is Logged OK and your messageId would be undefined. Btw, messageId is in your example not a number – Stramski Jan 21 at 21:15

`ApiMessages.logged_ok === 'Logged OK'` . in your function `message` is the string you want to send as the message. `ApiMessages.logged_ok` is the actual value of enum already! – Tadhg McDonald-Jensen Apr 26 at 20:11

1 Answer

12

As described in the handbook:

> Keep in mind that string enum members do not get a reverse mapping generated at all.

That means there is no simple reverse mapping in your case.

Workaround: Getting a reverse mapping for string enum members

To get the key of an enum member by it's value, you have to iterate through the enum keys and compare the associated value with your target value.

```
function getEnumKeyByEnumValue(myEnum, enumValue) {
    let keys = Object.keys(myEnum).filter(x => myEnum[x] == enumValue);
    return keys.length > 0 ? keys[0] : null;
}
```

Some demo code follows. You can also see it in action on the TypeScript Playground

```
enum ApiMessages {
    logged_ok = 'Logged OK',
    register_ok = 'Register OK'
}
```

```
alert(`The value '${exampleValue}' has the key '${exampleKey}'`)


function getEnumKeyByEnumValue(myEnum, enumValue) {
    let keys = Object.keys(ApiMessages).filter(x => myEnum[x] == enumValue);
    return keys.length > 0 ? keys[0] : null;
}
```

Adding this into your `responseOK()` you end up with:

```
function responseOK(message: ApiMessages, result ?: any) {
    return {
        "status": "ok",
        "code": 200,
        "messageId": getEnumKeyByEnumValue(ApiMessages, message),
        "message": message,
        "result": result
    };
}
```

edited Mar 19 at 9:57

answered Jan 21 at 21:14

Søren D. Ptæus
1,284 16 23

Oh, good to know. Thank you very much for the detailed answer and the demo. The workaround works like a charm. – miqrc Jan 21 at 21:32