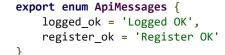
## Getting the enum key with the value string (reverse mapping) in TypeScript



I have an enum:







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
    };
}</pre>
```

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

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



100 132

asked Jan 21 at 20:31



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



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'
}
```

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

```
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



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

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