Accessing Enum Values

In this lesson, we will see how to access information from an enum.

We'll cover the following

- TypeScript map objects to allow access
- The JavaScript output

TypeScript map objects to allow access#

A variable set with an `enum` that has a `number` lets you access the `enum` name from the integer. However, an `enum` with string values does not have this capability. This means you can use the `enum` name followed by the name of the constant to get the value. Also, with a number, you can also use the value to return the name.

For example, an `enum` called `Orientation` with `East`, `West`, `North`, `South` could use `Orientation.East` to get the value zero or use `Orientation[0]` to get `East`. This works because TypeScript generates a map object which gives you access using the name of the entry *or* the value.

Here is the generated code of the orientation `enum`:

```
enum Orientation {
    East,
    West,
    North,
    South,
}
let directionInNumber = Orientation.East; // Access with the Enum
let directionInString = Orientation[0];  // Access the Enum string from number
console.log(directionInNumber);
console.log(directionInString);
```

As mentioned, it is not possible with an enum that has strings for value. The following code does not compile because lines 8 and 9 wrongly accessed the enum.

```
enum OrientationString {
    East = "E",
    West = "W",
    North = "N",
    South = "S",
}
let directionInNumber = OrientationString.East; // Access with the Enum
let directionInString = OrientationString[0];  // Access the Enum string from number
let directionInString2 = OrientationString["E"];  // Access the Enum string from number
console.log(directionInNumber);
console.log(directionInString);
console.log(directionInString2);
```

The JavaScript output#

The JavaScript output looks like the following for the first valid example:

```
let Orientation;
(function (Orientation) {
    Orientation[Orientation["East"] = 0] = "East";
    Orientation[Orientation["West"] = 1] = "West";
    Orientation[Orientation["North"] = 2] = "North";
    Orientation[Orientation["South"] = 3] = "South";
    Orientation[Orientation["South"] = 3] = "South";
})(Orientation || (Orientation = {}));
let directionInNumber = Orientation.East;
let directionInString = Orientation[0];
console.table(Orientation);
```

The JavaScript code generated by TypeScript creates a closure that assigns to a variable (`Orientation`) the four possible values by number as well as with string. The `Orientation` variable is an array with eight elements. The code on line 10 added an output that demonstrates how the values are accessible either way.

← Back

Enum With and Without Values

Next →

Merging and Adding Functionality to E...

✓ Mark as Completed

Report an Issue