Theory: Null and Undefined

6 minutes 0 / 5 problems solved

Skip this topic Start practicing

1360 users solved this topic. Latest completion was about 2 hours ago.

In JavaScript, there are two special data types to indicate the absence of a value: `null` and `undefined`. One might think, why would there be special data types to indicate what is not there, and why two of them? Well, in JS, things are not so simple. In this topic, we will figure out the difference between `null` and `undefined`, and learn when they are used.

§1. *null*

`null` means that the variable was explicitly assigned an empty or non-existent value. If the variable is `null`, we know that it does not contain an acceptable number, string or other data type:

```
let name = null;
console.log(name); // null
```

The example above shows that the variable `name` is unknown or for some reason does not have a value.

§2. *undefined*

The `undefined` value is returned when a variable was declared, but its value wasn't set. Let's consider the following example:

```
let count;
console.log(count); // undefined
```

Here, as expected, the `undefined` value was output to the console.

`undefined` may also occur in cases when object properties do not exist:

```
let person = {
  age: 27
};

console.log(person.name); // undefined
```

In this case, `undefined` is returned because we tried to output to the console a property `name` that wasn't specified in our object.

But that's not all! The `undefined` value is also returned when a function has a missing parameter:

```
function getDetails(a) {
  console.log(a);
}

getDetails(); // undefined
```

In the example above, `undefined` is printed because the parameter `a` was missing in the written function.

§3. Check the data types

`null` and `undefined` literally mean "no value", but they have different data types:

```
console.log(typeof null);       // object
console.log(typeof undefined);  // undefined
```

You expected that type `null` would return `null`, didn't you?

Current topic:

Null and Undefined

Topic depends on:

× Introduction to Objects

× Functions

Topic is required for:

Type conversion

Table of contents:

↑ Null and Undefined
§1. *null*
§2. *undefined*
§3. Check the data types
§4. Conclusion
Feedback & Comments

You were right if you thought so. In fact, `null` is not an object, but a separate data type. This strange behavior is an officially recognized error in the language. The mistake has existed for several decades. The reason no one fixes it is that thousands of sites created in JavaScript depend on this kind of wrong behavior. Fixing this bug can lead to many sites and popular web applications crashing. That's why we have nothing else to do but remember this peculiarity of the language and take it into account when writing scripts. We will talk more about objects as a separate type of data next time.

§4. Conclusion

Good job! Now you know what `null` and `undefined` have in common. You also know about the differences between these values and their weird features. Try not to abuse these data types: use them wisely, and you can avoid many mistakes in your programs.

Report a typo

116 users liked this theory. **2** didn't like it. **What about you?**

Start practicing

Comments (1) Hints (0) Useful links (0) Show discussion