deeecode

COERCION

What does this mean in JavaScript?

JS

swipe left

deeecode

JS

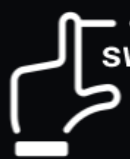
COERCION

Coercion is a concept in JavaScript values are converted from one data type to another, to fit an operation that you want to carry out. Coercion can be implicit (JavaScript automatically does it) or explicit (you do it yourself).

Let's look at **Implicit coercion**. Look at this code:

```
const sum = "20" + 30
```

Here, you are trying to add a string and a number together. Mathematically, this is not possible. Instead of throwing an error, JavaScript tries to "help" you.



swipe left

deeecode

JS

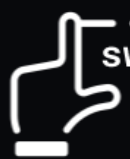
COERCION

JavaScript assumes that you made a mistake adding a number and a string, so what it does is, convert (coerce) the number to a string type. So JavaScript would execute the line of code like this:

```
const sum = "20" + "30"
// "2030"
```

The result is “2030” because **30** is converted to a string type and concatenated to **“20”**.

In the case of a plus sign (**+**), JavaScript only add if both values are numbers. If both aren’t numbers, JavaScript will do a concatenation, which in this case converts the number to a string.



swipe left

JS

COERCION

```
const times = "20" * 30
// 600
```

Here is another example where we use multiplication. Strings have nothing to do with the multiplication sign. Only numbers use it, so here, JavaScript converts the string to the number (if possible) before doing completing the operation.

If you try to use a string that cannot be converted to a number, you get **NaN**:

```
const times = "hello" * 30
// NaN
```

swipe left

JS

COERCION

Here's another example:

```
const sum = 50 + true
// 51
```

Here, we try to add a number **50** and a boolean **true**.

The number type is prioritized here, so **true** is converted to a number. The number representation of **true** is **1**, and **false** is **0**. You can try this in JavaScript by doing **Number(true)** or **Number(false)**.

Implicit coercion also occurs when you use the double equality:

swipe left

JS

COERCION

JavaScript has the **double** and **triple** equality operator. The double is called the **loose equality operator** while the triple is called the **strict equality operator**.

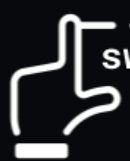
When you use the loose equality operator with values of different types, JavaScript could coerce in some cases.

Here is an example:

```
console.log("20" == 20)
// true
```

```
console.log(0 == false)
// true
```

“20” was coerced to a number type, so the equality returns true. false is coerced to a boolean, so true.



swipe left

deeecode

JS

COERCION

It is generally recommended to avoid implicit coercion. Becasue, this can cause inconsistencies and unknown bugs in your application.

When you mistakenly use values of different types for an operation, you should get an error and not some “help” from JavaScript.

To avoid implicit coercion, it is recommended that you use the strict equality (which doesn’t do a type conversion). Also, using type-checking tools like TypeScript helps you catch errors thrown using different data types together.

What about **explicit coercion**?

swipe left

deeecode

JS

COERCION

In some cases, you may intentionally want to convert a value from a data type to another. You can do this using Constructors, and other operators. Here are some examples:

```
// using constructors
Number("20") // 20
String(100) // "100"
Number(false) // 0
Boolean(20) // true
Boolean(0) // false
Boolean("hello") // true
```

```
// using operators
console.log(!100) // false
console.log(!!"hello") // true
console.log(+"40") // 40
```

If you enjoyed this post or have questions, do let me know in the comments.

Also, I'll putting up the video version on my YouTube soon. If you would love to see, check me out on YouTube...channel name is "**deeecode**".