JavaScript Type Conversion & Coercion | JavaScript Tutorial In Hindi #5

JavaScript Type Conversion & Coercion | JavaScript Tutorial In Hindi #5

In this tutorial, we will learn about type conversions and coercion in JavaScript with examples. JavaScript is a programming language used to build dynamic web pages. In this language, we do not have to specify the data type when we declare it. We can assign new data of a different type to the same variable. Under some circumstances, JavaScript will perform automatic type conversion. Typecasting/Type conversion and coercion in simple terms means changing the data type of a value to another data type like an integer to string, Boolean into String, etc.

JavaScript Type Conversion:-

When we convert one data type to another data type, this process is called type conversion. In JavaScript, there are two types of type conversion.

Explicit Conversion:-

The type conversion that we manually do is known as explicit type conversion. In JavaScript, explicit type conversions are done using the built-in methods like String (), Number(), etc.

- **Number Conversion:-**To convert Boolean values or numeric strings to numbers, we use the Number() i.e. an in-built method. Number() method in case of empty strings and null values return **0.** If a string is an invalid number like having an alphabet in a string, the result will be NaN.

The following are the rules of the numeric value:

Value	**Return**
undefined	NaN
null	0
true and false	1 and 0
string	Whitespaces from the start and end are removed. If the remaining string is empty, the result is 0. Otherwise, the number is

Value	Return
	"read" from the string. If the string contain any alphabet like 67a90, it will give NaN error.

For example:-

```
let res;
// string to number
res = Number('100');
console.log(res); // 100
// boolean to number
res = Number(false);
console.log(res); // 0
res = Number(' ')
console.log(res); // 0
res = Number('hello');
console.log(res); // NaN
res = Number(undefined);
console.log(res); // NaN
```

- **Boolean Conversion :-**Boolean type conversion happens in logical operations. It also follows the rules, but they are mostly obvious:
  - NaN, 0, undefined, null, "" are converting to false
  - everything else, including objects, to true

For Example:-

```
res=Boolean(1); // true
res= Boolean(0); // false
res= Boolean("hello"); // true
res= Boolean(""); // false
```

- **String Conversion:-**The method String() is used to convert numbers to strings. It can be used on any type of numbers, literals, variables, or expressions. The method toString() does the same.

Example:-

```
let res
let a=90
res= String(a) // returns a string from a number variable a
String(378009) // returns a string from a number literal 378009
a.toString()
(378009).toString()
```

- **parseInt and parseFloat:-** The in-built method parseInt() and parseFloat() return numbers from strings that start with numeric data. Here is an examples:

```
console.log( parseInt('$100 dollars') ); // NaN
console.log( parseInt('+10.25990 kg') ); // 10
console.log( parseFloat(' +10.25 kg') ); // 10.25
console.log( parseFloat('ABC’) ); //NaN
```

Implicit Conversion:-

Sometimes JavaScript automatically converts one data type to another. This type of conversion is known as implicit conversion.

- **Conversion To String:-** When we add a number into a string, JavaScript converts the number to a string before concatenation. Here is an example:

```
let res;
res = '3' + 4;
console.log(res) // "34"
res = '9' + true;
console.log(res); // "9true"
res= '0' + null;
console.log(res); // "0null"
```

- **Conversion To Number:-** Numeric string used with operators like +, - , / , * returns number type

```
let res
res = '4' - '4';
console.log(res); // 0
res = '4' * 5;
console.log(res); // 20
```

- **Boolean Conversion to Number:-** If we use Boolean, true is considered as 1 and false is considered as 0.

```
let res;
res = '5' - true;
console.log(res); // 4
res = 10 + false;
console.log(res); // 10
```

In JavaScript, Null is considered as 0 when used with numbers. Arithmetic operation of undefined with number, boolean or null returns NaN

```
res = 4 + null; // 4
res = 4 - undefined;// NaN
```

Code index.html as described/written in the video

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Tutorial on Js</title>
</head>
<body>
    <h1>This is Js tutorial by Harry</h1>
</body>

<!-- <script src="js/tut2.js"></script> -->
<!-- <script src="js/tut3.js"></script> -->
<!-- <script src="js/tut4.js"></script> -->
<!-- <script src="js/tut5.js"></script> -->
<!-- <script src="js/tut6.js"></script> -->
<!-- <script src="js/tut7.js"></script> -->
<!-- <script src="js/tut8.js"></script> -->
<!-- <script src="js/tut9.js"></script> -->
<!-- <script src="js/tut10.js"></script> -->
<script src="js/tut11.js"></script>
</html>
```

Js code as described/written in the video

Copy

```javascript
// Type conversion
console.log('Welcome to tut5');
let myVar;
myVar = String(34);
// console.log(myVar, (typeof myVar));
let booleanVar = String(true);
// console.log(booleanVar, (typeof booleanVar));

let date = String(new Date());
// console.log(date, (typeof date));
```

```
let arr =  String([1,2,3,4,5]);
// console.log(arr.length, (typeof arr));

let i = 75;
// console.log(i.toString())

let stri = Number("3434");
stri = Number("343d4");
stri = Number(false);
stri = Number([1,2,3,4,5,6,7,8,9]);
// console.log(stri, (typeof stri));

let number = parseFloat('34.098');


console.log(number.toFixed(2), (typeof number));

// Type coercion

let mystr = Number("698");
let mynum = 34;

console.log(mystr + mynum);
```

Previous Next