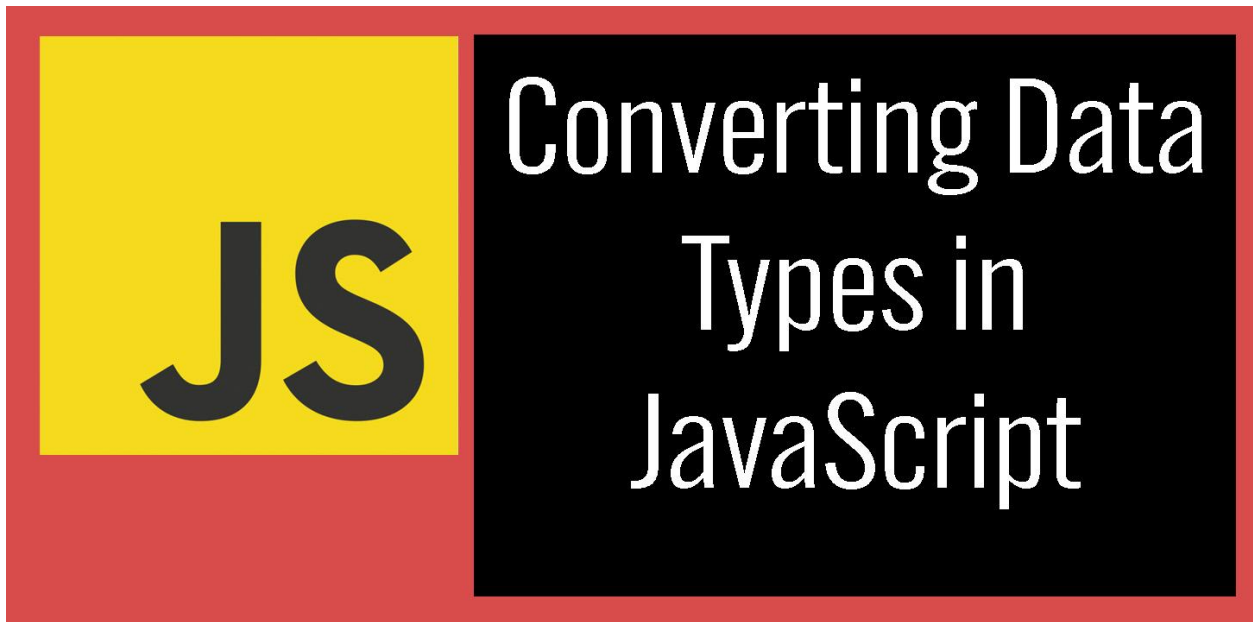


JavaScript Converting data types



1. Implicit Type Conversion (Type Coercion):

[Example 1: Implicit Conversion of Numbers to Strings](#)

[Example 2: Implicit Conversion of Strings to Numbers](#)

2. Explicit Type Conversion (Type Casting):

[Example 3: Converting Strings to Numbers](#)

[Example 4: Converting Numbers to Strings](#)

[Example 5: Converting Between Number Types](#)

[Example 6: Converting to Boolean](#)

Converting data types is a fundamental concept in JavaScript and many other programming languages. It involves changing the type of data from one form to another, which is often necessary to perform operations or comparisons. In JavaScript, you can convert data types primarily through implicit and explicit type conversion. Let's explore these concepts with coding examples:

Learn more about JavaScript with Examples and Source Code Laurence Svekis
Courses <https://basescripts.com/>

1. Implicit Type Conversion (Type Coercion):

JavaScript can automatically convert data types when needed. This is called implicit type conversion or type coercion. It typically occurs in operations involving different data types, like strings and numbers.

Example 1: Implicit Conversion of Numbers to Strings

```
let num = 42;  
let str = "The answer is: " + num; // Implicitly converts `num` to a string  
console.log(str); // Output: "The answer is: 42"
```

In this example, the number 42 is implicitly converted to a string when concatenated with the string.

Example 2: Implicit Conversion of Strings to Numbers

```
let strNum = "42";  
let num = strNum * 2; // Implicitly converts `strNum` to a number  
console.log(num); // Output: 84
```

Here, the string "42" is implicitly converted to a number when used in a mathematical operation.

2. Explicit Type Conversion (Type Casting):

Explicit type conversion, also known as type casting, allows you to manually convert data from one type to another using functions or methods provided by JavaScript.

Example 3: Converting Strings to Numbers

```
let strNum = "42";  
let num = parseInt(strNum); // Explicitly convert `strNum` to an integer  
console.log(num); // Output: 42
```

In this case, the `parseInt` function is used to explicitly convert the string "42" to an integer.

Example 4: Converting Numbers to Strings

```
let num = 42;  
let str = num.toString(); // Explicitly convert `num` to a string  
console.log(str); // Output: "42"
```

Here, the `toString` method is used to explicitly convert the number 42 to a string.

Example 5: Converting Between Number Types

```
let floatNum = 42.5;
```

```
let intNum = Math.floor(floatNum); // Explicitly convert a floating-point number to
an integer
console.log(intNum); // Output: 42
```

In this example, the `Math.floor` function is used to explicitly convert a floating-point number to an integer.

Example 6: Converting to Boolean

```
let truthyValue = "Hello";
let isTruthy = Boolean(truthyValue); // Explicitly convert to a boolean
console.log(isTruthy); // Output: true
```

```
let falsyValue = 0;
let isFalsy = Boolean(falsyValue); // Explicitly convert to a boolean
console.log(isFalsy); // Output: false
```

In these examples, the `Boolean` function is used to explicitly convert values to booleans.

Understanding and effectively using data type conversion is crucial in JavaScript to ensure that your code behaves as expected when working with different types of data. It's important to be mindful of both implicit and explicit conversions to avoid unexpected results and bugs in your code.