Type Coercion in JavaScript

Type coercion in JavaScript is the automatic or implicit conversion of values from one data type to another during operations.

This happens because JavaScript is a weakly typed language, which allows it to handle expressions involving mismatched types without throwing an error.

Type coercion can be categorized into two main types:

Implicit Coercion: JavaScript automatically converts the data types behind the scenes to perform an operation.

Explicit Coercion: The developer intentionally converts the data type using built-in functions like Number(), String(), or Boolean(). (this is type conversion and has already been covered)

How Implicit Coercion Works:

JavaScript follows specific rules for implicit coercion in different contexts.

Arithmetic Operations:
Most arithmetic operators (except +) will convert operands to numbers.

'12' - 2 results in 10 (number), because the string '12' is converted to the number 12 before subtraction.
'5' * '2' results in 10 (number).

The + operator has a special rule: if either operand is a string, the other is converted to a string, and concatenation occurs.

'1' + 2 results in '12' (string).
1 + 2 + 'number' results in '3number' (string), as the addition happens first, then string concatenation.

Boolean Contexts (Truthy and Falsy):
When a value is used in a logical context, such as an if statement or with logical operators (&&, ||, !), it is coerced to a boolean.
JavaScript has a small list of falsy values that convert to false. Everything else is truthy:

false
0 and -0
"" (empty string)
null
undefined
NaN

Examples:

Boolean(0) results in false.
if ('hello') the condition is true because a non-empty string is truthy.

Comparison Operations (==):
The loose equality operator (==) performs type coercion if the operands are of different types before comparing them. The strict equality operator (===), however, checks both the value and the type without any coercion, which is generally recommended to avoid unexpected bugs.
Examples:

12 == '12' results in true ('12' is coerced to the number 12).
false == 0 results in true (false is coerced to the number 0).
null == undefined results in true.
12 === '12' results in false (different types).

** In general, be aware of the string/number/Boolean operations, and do not let them happen by accident. If something doesn’t work the way it should, then perform an explicit conversion.*