ScopeTreasure Courses

JAVASCRIPT COURSES · LESSON 3.3 · CODE SCOPE SERIES

Data Types

The kinds of values JavaScript knows: number, string, boolean, undefined, null and object — typeof, and the famous "5" + 2 = "52" trap with its Number() fix.

Introduction

Every value in JavaScript has a **type** — a kind. A price is a different kind of thing from a name, and JavaScript treats them differently. If you treat text like a number, you get one of the most famous beginner surprises in all of programming: `"5" + 2` becomes `"52"`.

In this lesson you will meet the main types — **number, string, boolean, undefined, null,** and a first look at **object** — and you will learn to ask any value its type with `typeof`. By the end, the `"52"` trap will never catch you, because you will know the `Number()` fix.

(Roman Urdu: Har value ki aik qisam hoti hai — number, text, ya true/false — aur qisam galat ho to jawab bhi galat aata hai.)

Key points

- **number** — all numbers are one type: `250` and `99.5` are both numbers. JavaScript has **no separate** integer and decimal types.
- **string** — text inside quotes, single or double: `"Chai"` and `'Chai'` are both strings. Joining strings with `+` sticks them together: `"Chai" + "Point"` gives `"ChaiPoint"`. And `length` counts the characters: `"Chai".length` is `4`.
  (Roman Urdu: Quotes ke andar wali cheez text hai — number nahi, chahe woh "5" hi kyun na ho.)
- **boolean** — only two values: `true` and `false`, like a switch ON or OFF. Booleans answer yes/no questions in your code.
- **undefined** — a box was made but **nothing was put in yet**. JavaScript itself gives this value when a variable has no value.
- **null** — **deliberately empty**. The programmer sets `null` on purpose to say "nothing here, by my choice". Undefined = not filled yet; null = emptied on purpose.
  (Roman Urdu: undefined matlab abhi kuch rakha hi nahi; null matlab jaan boojh kar khaali rakha hai.)
- **object** — a bundle of named values, like an order: `order.price` reads the `price` inside the `order` object. Full objects lesson is 3.9.
- **`typeof` tells the type live:** `typeof 250` is `"number"`, `typeof "Chai"` is `"string"`. But beware the famous bug: `typeof null` is `"object"` — a mistake from 1995 that JavaScript can never fix, because too much old code depends on it.
- **The trap:** form values always arrive as **strings**. So `"5" + 2` is `"52"` — `+` joins text instead of adding. The fix: `Number("5") + 2` is `7`.

Code examples

Example 1 — Meet the types with typeof

```
let price = 250;
let shopName = "ChaiPoint";
let inStock = true;
let discount;
let coupon = null;

console.log(typeof price);
console.log(typeof shopName);
console.log(typeof inStock);
console.log(typeof discount);
console.log(coupon);
console.log(typeof null);
```

What it does: Five variables, five stories. `discount` was declared but given nothing, so it is `undefined`. `coupon` was set to `null` on purpose. The last line shows the famous 1995 bug with your own eyes.

What you will see: six lines in the Console: `number`, `string`, `boolean`, `undefined`, `null`, and — for `typeof null` — `object`.

Example 2 — Strings join, they do not add

```
let first = "Chai";
let second = "Point";

console.log(first + second);
console.log(first.length);
console.log(second.length);
```

What it does: The `+` between two strings **joins** them with no space in between. Then `length` counts characters: `"Chai"` has 4 letters and `"Point"` has 5.

What you will see: three lines in the Console: `ChaiPoint`, then `4`, then `5`.

Example 3 — The "52" trap and the Number() fix

```
let cupsFromForm = "5";

let wrong = cupsFromForm + 2;
console.log(wrong);
console.log(typeof wrong);

let right = Number(cupsFromForm) + 2;
console.log(right);
console.log(typeof right);
```

What it does: This is exactly what happens with real form input — the value `"5"` is a string, even though it looks like a number. Adding `2` joins it into `"52"`. Wrapping it in `Number()` converts the string into a real number first, so the maths works.

What you will see: four lines in the Console: `52`, then `string`, then `7`, then `number`. Same input — the type made all the difference.

Quick recap

- Main types: **number, string, boolean, undefined, null, object** — ask any value with `typeof`.
- `undefined` = nothing put in yet; `null` = emptied on purpose. And `typeof null` is `"object"` — the unfixable 1995 bug.
- Form values are strings: `"5" + 2` is `"52"`. Convert first with `Number()` and the maths is correct.

Common mistakes

1. **Adding form values without converting** — `"5" + 2` gives `"52"` because `+` joins strings. Fix: wrap input in `Number()` before doing maths.
2. **Mixing up undefined and null** — expecting `null` from a variable that was never given a value; it will be `undefined`. Fix: remember — JavaScript gives `undefined`; you give `null` on purpose.
3. **Trusting typeof null** — `typeof null` says `"object"`, which is a known bug, not the truth. Fix: check for null directly with `value === null`.

Practice

1. Create one variable of each type — number, string, boolean, undefined, and null — and print each value with its `typeof` result on one line each.
2. Take the string `"10"`, first add `5` to it and print the result, then use `Number()` and add `5` again. Print both results and say in one sentence why they differ.

Practice quiz

1. What does typeof null give, and why?

- "null", correctly
- "object", a famous 1995 bug JavaScript can never fix
- "undefined"
- An error

Answer: "object", a famous 1995 bug JavaScript can never fix

2. What is "5" + 2 in JavaScript?

- 7
- "52", because + joins strings and form values arrive as strings
- 10
- NaN

Answer: "52", because + joins strings and form values arrive as strings

3. What is the difference between undefined and null?

- They are identical
- undefined means nothing was put in yet (JavaScript gives it), null means deliberately emptied on purpose
- null means nothing put in yet, undefined means emptied on purpose
- undefined is for numbers, null is for text

Answer: undefined means nothing was put in yet (JavaScript gives it), null means deliberately emptied on purpose

---

Watch the video lesson: https://youtu.be/1SVyNEX_YWI

Read online: https://scopetreasure.com/notes/js-data-types · More free tools and notes at https://scopetreasure.com