Values, variables & types

Ref Javascript

let and const

A variable is a named box that holds a value. Make one with `let` or `const`, then use its name:

```
let score = 10;
score = score + 5;      // let: the value can change
console.log(score);     // 15

const name = "Mei";     // const: the name cannot be reassigned
console.log(name);      // Mei
```

- = means **assign** : work out the right side, then store it in the box on the left.
- Use `const` by default. Use `let` only when the value must change later.
- Reassigning a `const` is an error: `name = "Tom"` is not allowed. A const object or array can still have its contents changed.

Numbers & maths

These examples use the number type for whole and decimal values. BigInt is a separate integer type. The arithmetic operators are:

Operator	Meaning	Example	Result
+ -	add, subtract	7 - 2	5
* /	multiply, divide	7 / 2	3.5
%	remainder	7 % 2	1
**	power	2 ** 3	8

```
console.log(7 / 2);               // 3.5
console.log(Math.floor(7 / 2));   // 3   (whole-number division)
console.log(7 % 2);               // 1   (odd numbers leave 1)
console.log(2 + 3 * 4);           // 14  (* before +)
console.log((2 + 3) * 4);         // 20
```

`Math` holds useful functions:

```
console.log(Math.round(2.5));     // 3
console.log(Math.max(4, 9, 2));   // 9
console.log(Math.sqrt(49));       // 7
```

- `Math.random()` gives a random number from 0 up to (not including) 1. `Math.floor(Math.random() * 6) + 1` is a dice roll from 1 to 6.
- Decimals are stored in binary, so `0.1 + 0.2` prints `0.30000000000000004`. Round a result before you show it: `(0.1 + 0.2).toFixed(2)` is `"0.30"`.

Types & conversion

Every value has a type. `typeof` tells you which:

```
console.log(typeof 42);          // number
console.log(typeof "42");        // string
console.log(typeof true);        // boolean
console.log(typeof undefined);   // undefined
```

- A variable made with `let x;` and no value is `undefined`.
- `null` means "no value", set on purpose.

`+` with a string **joins** instead of adding, so convert text to a number first:

```
console.log("5" + 1);            // 51  (joined)
console.log(Number("5") + 1);    // 6   (added)
console.log(parseInt("12px"));   // 12  (reads the number at the start)
console.log(String(9) + "!");    // 9!
```

- A text that is not a number gives `NaN` ("not a number"): `Number("abc")` is `NaN`.
- `-`, `*` and `/` convert text to a number by themselves (`"6" * 2` is `12`), but do not rely on it: convert on purpose.

Common mistakes

- `=` assigns; it does not compare. Comparing uses `===` (topic 4).
- `"5" + 1` is `"51"`. Convert input with `Number(...)` before doing maths with it.
- `Math.round(2.5)` is `3`, but `Math.round(-2.5)` is `-2`: it rounds halves up, towards the larger number.