JAVASCRIPT:

Template Literals | String Methods & Properties

JavaScript Strings & Useful Methods (Short & Best Notes)

✅ 1. What is a String?

A sequence of characters enclosed in `""`, `''`, or `` ` ` `` **(backticks)**.

```
javascript
CopyEdit
let str1 = "Hello";
let str2 = 'World';
let str3 = `JavaScript`;
```

✅ 2. String Properties:

- **Length** → `str.length` → Returns string length.

  ```
  console.log("Hello".length); // 5
  ```

✅ 3. Important String Methods:

Method	Use & Example
`toUpperCase()`	`"hello".toUpperCase()` → `"HELLO"`
`toLowerCase()`	`"HELLO".toLowerCase()` → `"hello"`
`trim()`	`"  hello  ".trim()` → `"hello"`
`charAt(index)`	`"Hello".charAt(1)` → `"e"`
`indexOf("word")`	`"Hello".indexOf("e")` → `1` (First occurrence)
`lastIndexOf("word")`	`"Hello Hello".lastIndexOf("o")` → `10`
`includes("word")`	`"Hello".includes("ll")` → `true`
`startsWith("word")`	`"Hello".startsWith("He")` → `true`

Method	Use & Example
`endsWith("word")`	`"Hello".endsWith("lo")` → `true`
`slice(start, end)`	`"Hello".slice(1, 4)` → `"ell"`
`substring(start, end)`	`"Hello".substring(1, 4)` → `"ell"` (Similar to `slice`)
`replace("old", "new")`	`"Hello".replace("H", "Y")` → `"Yello"`
`replaceAll("old", "new")`	`"aaabbb".replaceAll("a", "x")` → `"xxxbbb"`
`split("separator")`	`"Hello World".split(" ")` → `["Hello", "World"]`
`repeat(n)`	`"Hi".repeat(3)` → `"HiHiHi"`

✅ 4. String Concatenation:

```
let name = "John";
console.log("Hello " + name); // Using +
console.log(`Hello ${name}`); // Using Template Literals
```

📌 `slice()` vs `substring()` in JavaScript (Short & Clear)

Both extract part of a string, but they have **small differences**.

✅ 1. `slice(start, end)`

- **Start index is included, End index is excluded** (`end` is not taken).
- **Accepts negative indexes** (counts from the end).

```
let str = "JavaScript";
console.log(str.slice(1, 4));   // "ava" (takes index 1 to 3)
console.log(str.slice(-6, -1)); // "Scrip" (Negative index works)
```

✅ 2. `substring(start, end)`

- **Start index is included, End index is excluded** (`end` is not taken).
- **Does NOT accept negative indexes** (negative values are treated as `0`).
- If `start > end`, it **swaps** them automatically.

```
let str = "JavaScript";
console.log(str.substring(1, 4));   // "ava" (same as slice)
console.log(str.substring(4, 1));   // "ava" (Swaps start & end)
console.log(str.substring(-6, 4));  // "Java" (Negative treated as 0)
```

📌 Key Differences:

Feature	`slice()` ✅	`substring()` ✅
Accepts negative indexes?	✅ Yes	❌ No (treated as 0)
Swaps `start` & `end` if `start > end`?	❌ No	✅ Yes

🖍️ **Best Practice:**

- Use **`slice()`** when working with negative indexes.
- Use **`substring()`** if you want automatic swapping.