⬛ LECTURE 6 — STRINGS IN JAVASCRIPT

📘 **JavaScript Core — Strings (Text Handling Mastery)**
🧠 *First Principles • Real-Life Examples • Complete Toolkit*

⬛ FIRST PRINCIPLE — "STRING HOTI KYA HAI?"

🔷 Simple Definition

String = characters ka **sequence (text)**
Jo hamesha quotes ke andar likha jata hai.

👉 JavaScript me strings likhne ke **3 tareeke** hote hain:

- `'single quotes'`
- `"double quotes"`
- `` `backticks` ``

🟨 **Real-Life Analogy**

WhatsApp message, username, comment → **STRING**

⬛ WAYS TO DECLARE STRING

```
let singleQuote = 'Hello, world';

let doubleQuote = "Hello, world";

let backticks   = `Hello, world`;
```

🟨 **IMPORTANT**

Teenon ka kaam same hai —
par **backticks** extra power dete hain

⬛ STRING INTERPOLATION (BACKTICKS MAGIC)

🔷 Problem (Without Backticks)

String ke andar variable jodna messy ho jata hai.

🔷 Solution → Template Literals ( )

```
let price = 80;

console.log(`Price of the tomato is ${price}`);
```

🟨 OUTPUT

```
Price of the tomato is 80
```

🟨 RULE

`${variable}` sirf **backticks** ke andar kaam karta hai

⬛ STRING CONCATENATION

🔷 Meaning

Do ya zyada strings ko **jodna**

```
let s1 = "hello";

let s2 = " world";

let s3 = s1 + s2;


console.log(s3); // "hello world"
```

🟨 NOTE

`+` operator string ke sath → **concatenation**

⬛ LENGTH OF STRING

```
console.log(s1.length); // 5

console.log(s2.length); // 6

console.log(s3.length); // 11
```

🟨 **RULE**

`.length` property
→ total characters count karta hai (spaces bhi)

⬛ ACCESSING CHARACTERS

```
let special = "Rohit";


console.log(special[0]);        // 'R'

console.log(special.charAt(3)); // 'i'
```

🟨 **DIFFERENCE**

- `str[index]` → modern & common
- `charAt()` → safe (out of range pe empty string)

⬛ CHANGE CASE (UPPER / LOWER)

```
console.log(special.toLowerCase()); // 'rohit'

console.log(special.toUpperCase()); // 'ROHIT'
```

🟨 **IMPORTANT**

Strings **immutable** hoti hain
Original string change nahi hoti

⬛ SEARCHING IN STRINGS

🔷 Important Search Methods

- `indexOf()` → first occurrence
- `lastIndexOf()` → last occurrence
- `includes()` → true / false

```
let hero = "Hello Coder Coder Army";


console.log(hero.indexOf("Coder"));     // 6
console.log(hero.lastIndexOf("Coder")); // 12
console.log(hero.includes("Coder"));    // true
```

🟥 **RULE**

Agar text nahi mila → `indexOf()` returns `-1`

⬛ EXTRACTING SUBSTRINGS

🔷 slice vs substring

Method	Negative Index
`slice()`	✅ Allowed
`substring()`	❌ Not allowed

```
let newstr = "HelloDon";


console.log(newstr.slice(0, 3));       // 'Hel'
console.log(newstr.substring(0, 3));   // 'Hel'
console.log(newstr.slice(-6, -3));     // 'llo'
console.log(newstr.slice(-2, 4));      // ''
```

🟨 **TIP**

Negative index chahiye → **slice() use karo**

⬛ REPLACING CONTENTS

```
let str10 = "hello world world acha hai";


console.log(str10.replace("world", "duniya"));

// hello duniya world acha hai


console.log(str10.replaceAll("world", "duniya"));

// hello duniya duniya acha hai
```

🟨 **RULE**

- `replace()` → first match
- `replaceAll()` → all matches

⬛ SPLITTING STRINGS

```
let str2 = "money, honey, sunny, funny";

console.log(str2.split(", "));
```

🟨 **OUTPUT**

```
['money', 'honey', 'sunny', 'funny']
```

🟨 **RULE**

`split()` → **string ko array me convert karta hai**

⬛ TRIMMING STRINGS (WHITESPACE REMOVE)

```
let str3 = "   Coder Army   ";


console.log(str3.trim());      // "Coder Army"
```

```
console.log(str3.trimStart()); // "Coder Army   "

console.log(str3.trimEnd());   // "   Coder Army"
```

🟨 **Real-Life Use**

User input clean karne ke liye

⬛ EXTRA USEFUL STRING METHODS

```
let text = "JavaScript is fun";


console.log(text.startsWith("Java")); // true

console.log(text.endsWith("fun"));    // true

console.log(text.repeat(2));          // repeat string
```

Padding (Mostly IDs / OTPs)

```
console.log("7".padStart(3, "0")); // "007"

console.log("7".padEnd(3, "0"));   // "700"
```

⬛ STRING OBJECT vs STRING PRIMITIVE

```
let latestString = new String("hello coder army");

console.log(typeof latestString); // object
```

🟥 **IMPORTANT DIFFERENCE**

- `"hello"` → primitive → stack
- `new String()` → object → heap

🟨 **BEST PRACTICE**

❌ `new String()` avoid karo
✅ Primitive string use karo

⬛ SHORT SUMMARY — EK NAZAR ME ⬛

✅ Declare → `' '`, `" "`, `` ` ` ``
✅ Interpolation → `${}` (backticks only)
✅ Concatenation → `+`
✅ Length → `.length`
✅ Access → `str[i]`, `charAt()`
✅ Case → `toUpperCase()`, `toLowerCase()`
✅ Search → `indexOf`, `lastIndexOf`, `includes`
✅ Extract → `slice`, `substring`
✅ Replace → `replace`, `replaceAll`
✅ Split → `split()`
✅ Trim → `trim`, `trimStart`, `trimEnd`
✅ Extra → `startsWith`, `endsWith`, `repeat`, `padStart`, `padEnd`
✅ New String → object (avoid)

⬛ FINAL THOUGHT ⬛

🧠 **Strings = JavaScript ka sabse zyada used data type**
Isko master kar liya →
👉 **Forms, APIs, UI, validation sab easy**
