Yasir

WEEK 2 : DAY 3

STRING DATA TYPE

Yasir

INTRODUCTION TO STRINGS

Strings are one of the most commonly used data types in JavaScript. They are used to represent and manipulate text.

- Creating Strings
- String Methods
- Template Literals
- String Concatenation
- Escape Sequences
- String Immutability

Yasir

CREATING STRINGS

There are three ways to create strings in JavaScript:

```
// Single Quotes
let single = 'single quotes';

// Double Quotes
let double = "double quotes";

// Backticks
let backticks = `backticks`;
```

Template literals allow for embedding variables and expressions:

```
let name = 'John';
let age = 30;
let sentence = `My name is ${name} and I am ${age} years old.`;
```

Yasir

20 COMMON STRING METHODS

JavaScript provides a variety of methods to manipulate strings:

length property

```
let str = 'hello';
let length = str.length; // 5
```

charAt()

```
let char = str.charAt(0); // 'h'
```

concat()

```
let str1 = 'hello';
let str2 = 'world';
let combined = str1.concat(' ', str2); // 'hello world'
```

includes()

```
let str3 = 'hello world';
let included = str3.includes('world'); // true
```

indexOf()

```
let str5 = 'hello world';
let lastIndex = str5.lastIndexOf('o'); // 7
```

Yasir

CONTINUE...

replace()

```
let str6 = 'hello world';
let replaced = str6.replace('world', 'there'); // 'hello there'
```

slice()

```
let str7 = 'hello world';
let sliced = str7.slice(6); // 'world'
```

substring()

```
let str9 = 'hello world';
let substring = str9.substring(6); // 'world'
```

toUpperCase()

```
let str10 = 'hello';
let upper = str10.toUpperCase(); // 'HELLO'
```

toLowerCase()

```
let str11 = 'HELLO';
let lower = str11.toLowerCase(); // 'hello'
```

Yasir

CONTINUE...

trim()

```
let str12 = '    hello    ';
let trimmed = str12.trim(); // 'hello'
```

split()

```
let str13 = 'hello world';
let splitted = str13.split(' '); // ['hello', 'world']
```

repeat()

```
let str14 = 'hello';
let repeated = str14.repeat(3); // 'hellohellohello'
```

match()

```
let str15 = 'hello world';
let matched = str15.match(/o/g); // ['o', 'o']
```

toString()

```
let str17 = 123;
let string = str17.toString(); // '123'
```

Yasir

TEMPLATE LITERALS

Template literals are a new way to create strings in JavaScript. They allow for embedding variables and expressions without concatenation.

Multiline Strings: Use backticks to create strings that span multiple lines.

```
let multiline = `
  This is a
  multiline
  string.
`;
```

Embedding Variables

```
let name = 'John';
let age = 30;
let sentence = `My name is ${name} and I am ${age}
years old.`; // 'My name is John and I am 30 years
old.'
```

Yasir

STRING CONCATENATION

Concatenation combines two or more strings into one:

Using + Operator:

```
let str1 = 'hello';
let str2 = 'world';
let combined = str1 + ' ' + str2; // 'hello world'
```

Using concat() Method:

```
let str1 = 'hello';
let str2 = 'world';
let combined = str1.concat(' ', str2); // 'hello wo
rld'
```

Using Template Literals:

```
let str1 = 'hello';
let str2 = 'world';
let combined = `${str1} ${str2}`; // 'hello world'
```

Yasir

ESCAPE SEQUENCES

Escape sequences allow you to include special characters in strings:

- New Line: \n - Adds a new line.
- Tab: \t - Adds a tab.
- Backslash: \\ - Adds a backslash.
- Quotes: \' and \" - Adds single or double quotes.

```
// New Line: \n - Adds a new line.
let newline = 'hello\nworld';
// Tab: \t - Adds a tab.
let tab = 'hello\tworld';
// Backslash: \\ - Adds a backslash.
let backslash = 'hello\\world';
// Quotes: \' and \" - Adds single or double quotes.
let quotes = 'hello\'world';
let doubleQuotes = 'hello\"world';

```

Yasir

STRING IMMUTABILITY

Strings in JavaScript are immutable, meaning they cannot be changed once created. Any method that modifies a string actually returns a new string.

When you modify a string, a new string is created instead of altering the original string.

```
let str = 'Hello';
str[0] = 'h'; // This will not change the string.
str = 'hello'; // This will create a new string.
```

Yasir

WHAT'S NEXT?

NUMBER DATA TYPE (IN-DEPTH)

In our next session, we will delve into the Number Data Type in JavaScript. Numbers are another fundamental data type, essential for performing mathematical operations and handling numeric data.

Key Points we will Cover -

- Types of Numbers
- Number Properties
- Number Methods
- Mathematical Operations
- Math Object
- Type Conversion
- Common Pitfalls and Best Practices

Stay tuned for an in-depth exploration of numbers in JavaScript, where we'll uncover the power and nuances of numeric operations and how to handle them effectively in your programs!

Yasir

STAY CONNECTED

Thank you for joining me on this JavaScript journey! Your commitment to learning and growing is truly inspiring.

Stay tuned for more in-depth content as we dive deeper into JavaScript in the upcoming weeks. Keep coding and keep learning!

If you have any questions, feedback, or suggestions, please feel free to reach out. I'm here to help you succeed in your JavaScript journey.

Follow me here

github.com/yasirmansoori

yasirmansoori.tech