String in JavaScript?

JavaScript String is a sequence of characters, typically used to represent text. It is enclosed in single or double quotes.

String Creation: Strings can be created using single quotes ('), double quotes ("), or template literals (`).

let string1 = 'Hello, world!';

let string2 = "Hello, world!";

let string3 = `Hello, world!`;

Template Literals: Introduced in ES6 (ECMAScript 2015), template literals provide more capabilities than the older single and double quotes. They allow multiline strings and can embed expressions, which are evaluated and then converted into a resulting string.

let name = 'Alice';

let greeting = `Hello, ${name}!`;

console.log(greeting); // Hello, Alice!

Multiline Strings: Prior to ES6, strings went across multiple lines had to be concatenated using the + operator or by including newline characters (\n). Template literals simplify this.

let longString = `This is a very

long string that spans

multiple lines`;

Basic String Operations

charAt(index): Returns the character at the specified index.

charCodeAt(index): Returns the Unicode value of the character at the specified index.

concat(...strings): Combines the text of two or more strings and returns a new string.

includes(searchString, position): Determines whether one string may be found within another string, returning true or false as appropriate.

endsWith(searchString, length): Determines whether a string ends with the characters of a specified string, returning true or false as appropriate.

startsWith(searchString, position): Determines whether a string begins with the characters of another string, returning true or false as appropriate.

indexOf(searchValue, fromIndex): Returns the index within the calling String object of the first occurrence of the specified value, starting the search at fromIndex. Returns -1 if the value is not found.

lastIndexOf(searchValue, fromIndex): Returns the index within the calling String object of the last occurrence of the specified value, searching backwards from fromIndex. Returns -1 if the value is not found.

match(regexp): Used to match a regular expression against a string.

matchAll(regexp): Returns an iterator of all results matching a string against a regular expression, including capturing groups.

normalize(form): Returns the Unicode Normalization Form of the string.

String Generation and Formatting

repeat(count): Returns a string consisting of the elements of the object repeated the given times.

replace(searchFor, replaceWith): Used to replace occurrences within a string.

replaceWith(replacer): Replaces the results of a regular expression with another string.

replaceAll(searchFor, replaceWith): Used to replace all occurrences within a string.

slice(beginIndex, endIndex): Extracts a section of a string and returns it as a new string, without modifying the original string.

split(separator, limit): Splits a String object into an array of strings by separating the string into substrings.

substring(indexStart, indexEnd): Returns a subset of a string between one index and another, or through the end of the string.

substr(start, length): Returns the characters in a string beginning at the specified location through the specified number of characters.

Case Conversion Methods

toLowerCase(): Returns the calling string value converted to lower case.

toLocaleLowerCase(): The characters within a string are converted to lower case while respecting the current locale.

toUpperCase(): Returns the calling string value converted to upper case.

toLocaleUpperCase(): The characters within a string are converted to upper case while respecting the current locale.

Whitespace Handling

trim(): Trims whitespace from both ends of a string.

trimStart() or trimLeft(): Trims whitespace from the beginning of a string.

trimEnd() or trimRight(): Trims whitespace from the end of a string.

Miscellaneous Methods

valueOf(): Returns the primitive value of a string object.

toString(): Returns a string representing the specified object. Overrides the Object.prototype.toString() method.

padStart(targetLength, padString): Pads the current string from the start with another string (multiple times, if needed) until the resulting string reaches the given length.

padEnd(targetLength, padString): Pads the current string from the end with another string (multiple times, if needed) until the resulting string reaches the given length.

Each of these methods provides different ways to manipulate and handle strings, giving JavaScript developers powerful tools to process and modify textual data efficiently.

Basic String Operations

charAt(index)

javascript

Copy code

'hello'.charAt(1); // 'e'

charCodeAt(index)

javascript

Copy code

'hello'.charCodeAt(1); // 101

concat(...strings)

javascript

Copy code

'Hello'.concat(' ', 'world!'); // 'Hello world!'

includes(searchString, position)

javascript

Copy code

'hello world'.includes('world'); // true

endsWith(searchString, length)

javascript

Copy code

'hello world'.endsWith('world'); // true

startsWith(searchString, position)

javascript

Copy code

'hello world'.startsWith('hello'); // true

indexOf(searchValue, fromIndex)

javascript

Copy code

'hello world'.indexOf('o'); // 4

lastIndexOf(searchValue, fromIndex)

javascript

Copy code

'hello world'.lastIndexOf('o'); // 7

match(regexp)

javascript

Copy code

'hello world'.match(/[a-z]/g); // ['h', 'e', 'l', 'l', 'o', 'w', 'o', 'r', 'l', 'd']

matchAll(regexp)

javascript

Copy code

Array.from('test1 test2'.matchAll(/\w+/g)); // [['test1'], ['test2']]

normalize(form)

javascript

Copy code

'NFD'.normalize('NFC'); // 'NFD'

String Generation and Formatting

repeat(count)

javascript

Copy code

'abc'.repeat(3); // 'abcabcabc'

replace(searchFor, replaceWith)

javascript

Copy code

'hello world'.replace('world', 'everyone'); // 'hello everyone'

replaceWith(replacer)

javascript

Copy code

// As of my last update, replaceWith doesn't exist as a standard JavaScript String method. Let's skip this.

replaceAll(searchFor, replaceWith)

javascript

Copy code

'hello world world'.replaceAll('world', 'everyone'); // 'hello everyone everyone'

slice(beginIndex, endIndex)

javascript

Copy code

'hello world'.slice(6, 11); // 'world'

split(separator, limit)

javascript

Copy code

'a,b,c'.split(','); // ['a', 'b', 'c']

substring(indexStart, indexEnd)

javascript

Copy code

'hello world'.substring(0, 5); // 'hello'

substr(start, length)

javascript

Copy code

'hello world'.substr(1, 4); // 'ello'

Case Conversion Methods

toLowerCase()

javascript

Copy code

'HELLO WORLD'.toLowerCase(); // 'hello world'

toLocaleLowerCase()

javascript

Copy code

'HELLO WORLD'.toLocaleLowerCase(); // 'hello world'

toUpperCase()

javascript

Copy code

'hello world'.toUpperCase(); // 'HELLO WORLD'

toLocaleUpperCase()

javascript

Copy code

'hello world'.toLocaleUpperCase(); // 'HELLO WORLD'

Whitespace Handling

trim()

javascript

Copy code

' hello world '.trim(); // 'hello world'

trimStart() or trimLeft()

javascript

Copy code

' hello world '.trimStart(); // 'hello world '

trimEnd() or trimRight()

javascript

Copy code

' hello world '.trimEnd(); // ' hello world'

Miscellaneous Methods

valueOf()

javascript

Copy code

var strObj = new String('hello world');

strObj.valueOf(); // 'hello world'

toString()

javascript

Copy code

var strObj = new String('hello world');

strObj.toString(); // 'hello world'

padStart(targetLength, padString)

javascript

Copy code

'5'.padStart(3, '0'); // '005'

padEnd(targetLength, padString)

javascript

Copy code

'5'.padEnd(3, '0'); // '500'