**String :**

The **String** object is used to represent and manipulate a sequence of characters.

## **String properties**

**1. constructor**

The constructor property returns the function that created the String prototype.

## **Syntax**

*string*.constructor

### **Example**

let message = "Hello World!";  
let text = message.constructor;

**2. length**

The length property returns the length of a string.

The length property of an empty string is 0.

## **Syntax**

*string*.length

### **Example**

let text = "Hello World!";  
let length = text.length;

**3. prototype**

The prototype is a property available with all JavaScript objects.

The prototype property allows you to add new properties and methods to strings.

## **Syntax**

object.prototype.name = value

### **Example**

function employee(name, jobtitle, born) {  
  this.name = name;  
  this.jobtitle = jobtitle;  
  this.born = born;  
}  
employee.prototype.salary = 2000;  
const fred = new employee("Fred Flintstone", "Caveman", 1970);

## **String Methods**

**1. charAt()**

The charAt() method returns the character at a specified index (position) in a string.

The index of the first character is 0, the second 1, ...

The index of the last character is string length - 1

## **Syntax**

*string*.charAt(index)

### **Examples**

let text = "HELLO WORLD";  
let letter = text.charAt(0);

**2. charCodeAt()**

The charCodeAt() method returns the Unicode of the character at a specified index (position) in a string.

The index of the first character is 0, the second is 1, ....

## **Syntax**

*string*.charCodeAt(index)

### **Examples**

let text = "HELLO WORLD";  
let code = text.charCodeAt(0);

**3.concat()**

The concat() method joins two or more strings.

The concat() method does not change the existing strings.

The concat() method returns a new string.

## **Syntax**

*string*.concat(string1, string2, ..., stringX)

### **Examples**

let text1 = "sea";  
let text2 = "food";  
let result = text1.concat(text2);

**4.endWith()**

The endsWith() method returns true if a string ends with a specified string.

Otherwise it returns false.

The endsWith() method is case sensitive.

## **Syntax**

*string*.endsWith(searchvalue, length)

### **Examples**

let text = "Hello world";  
let result = text.endsWith("world");

## **5. includes() 6.indexOf() 7. lastIndexOf()**

## The includes() method returns true if a string contains a specified string.

Otherwise it returns false.

## **Syntax**

*string*.includes(searchvalue, start)

### **Examples**

let text = "Hello world, welcome to the universe.";  
let result = text.includes("world");

## **6.indexOf()**

The indexOf() method returns the position of the first occurrence of a value in a string.

The indexOf() method returns -1 if the value is not found.

The indexOf() method is case sensitive.

## **Syntax**

*string*.indexOf(searchvalue, start)

### **Examples**

let text = "Hello world, welcome to the universe.";  
let result = text.indexOf("welcome");

**7. lastIndexOf()**

The lastIndexOf() method returns the index (position) of the last occurrence of a specified value in a string.

The lastIndexOf() method searches the string from the end to the beginning.

The lastIndexOf() method returns the index from the beginning (position 0).

## **Syntax**

*string*.lastIndexOf(searchvalue, start)

### **Examples**

let text = "Hello planet earth, you are a great planet.";  
let result = text.lastIndexOf("planet");

**8. localeCompare()**

The localeCompare() method compares two strings in the current locale.

The localeCompare() method returns sort order -1, 1, or 0 (for before, after, or equal).

The current locale is based on the language settings of the browser.

## **Syntax**

*string*.localeCompare(compareString)

### **Examples**

let text1 = "ab";  
let text2 = "cd";  
let result = text1.localeCompare(text2);

**9.** **match()**

The match() method matches a string against a regular expression

The match() method returns an array with the matches.

The match() method returns null if no match is found.

## **Syntax**

*string*.match(match)

### **Examples**

let text = "The rain in SPAIN stays mainly in the plain";  
text.match("ain");

**10. replace()**

The replace() method searches a string for a value or a regular expression.

The replace() method returns a new string with the value(s) replaced.

The replace() method does not change the original string.

## **Syntax**

*string*.replace(*searchValue, newValue*)

### **Examples**

let text = "Visit Microsoft!";  
let result = text.replace("Microsoft", "W3Schools");

**11. Search()**

The search() method matches a string against a regular expression \*\*

The search() method returns the index (position) of the first match.

The search() method returns -1 if no match is found.

The search() method is case sensitive.

## **Syntax**

*string*.search(searchValue)

### **Examples**

let text = "Mr. Blue has a blue house";  
let position = text.search("Blue");

## **12. slice()**

The slice() method extracts a part of a string.

The slice() method returns the extracted part in a new string.

The slice() method does not change the original string.

## **Syntax**

*string*.slice(start, end)

### **Examples**

let text = "Hello world!";  
let result = text.slice(0, 5);

**13. Split()**

The split() method splits a string into an array of substrings.

The split() method returns the new array.

The split() method does not change the original string.

If (" ") is used as separator, the string is split between words.

## **Syntax**

*string*.split(separator, limit)

### **Examples**

let text = "How are you doing today?";  
const myArray = text.split(" ");

**14. startsWith()**

The startsWith() method returns true if a string starts with a specified string.

Otherwise it returns false.

The startsWith() method is case sensitive.

## **Syntax**

*string*.startsWith(searchValue, start)

### **Examples**

let text = "Hello world, welcome to the universe.";  
text.startsWith("Hello");

## **15. Substr()**

The substr() method extracts a part of a string.

The substr() method begins at a specified position, and returns a specified number of characters.

The substr() method does not change the original string.

## **Syntax**

*string*.substr(start, length)

### **Examples**

let text = "Hello world!";  
let result = text.substr(1, 4);"

## **16. substring()**

The substring() method extracts characters, between two indices (positions), from a string, and returns the substring.

The substring() method extracts characters from start to end (exclusive).

The substring() method does not change the original string.

## **Syntax**

*string*.substring(start, end)

### **Examples**

let text = "Hello world!";  
let result = text.substring(1, 4);

**17. toLocaleLowerCase()**

The toLocaleLowerCase() method converts a string to lowercase letters, using current locale.

The locale is based on the language settings of the browser.

The toLocaleLowerCase() method does not change the original string.

The toLocaleLowerCase() returns the same result as toLowerCase(), except for locales that conflict with the regular Unicode case mappings

## **Syntax**

*string*.toLocaleLowerCase()

### **Example**

let text = "Hello World!";  
let result = text.toLocaleLowerCase();

**18. toLocaleUpperCase()**

The toLocaleUpperCase() method converts a string to uppercase letters, using current locale.

The locale is based on the language settings of the browser.

The toLocaleUpperCase() method does not change the original string.

The toLocaleUpperCase() returns the same result as toUpperCase(), except for locales that conflict with the regular Unicode case mappings

## **Syntax**

*string*.toLocaleUpperCase()

### **Example**

let text = "Hello World!";  
let result = text.toLocaleUpperCase();

**19. toString()**

The toString() method returns a string as a string.

The toString() method does not change the original string.

The toString() method can be used to convert a string object into a string.

## **Syntax**

*string*.toString()

### **Example**

let text = "Hello World!";  
let result = text.toString();

**20. toUpperCase()**

The toUpperCase() method converts a string to uppercase letters.

The toUpperCase() method does not change the original string.

## **Syntax**

*string*.toUpperCase()

### **Example**

let text = "Hello World!";  
let result = text.toUpperCase();

**21. toLowerCase()**

The toLowerCase() method converts a string to lowercase letters.

The toLowerCase() method does not change the original string.

## **Syntax**

*string*.toLowerCase()

### **Example**

let text = "Hello World!";  
let result = text.toLowerCase();

**22. trim()**

The trim() method removes whitespace from both sides of a string.

The trim() method does not change the original string.

## **Syntax**

*string*.trim()

### **Example**

let text = "       Hello World!        ";  
let result = text.trim();

## **23. valueOf()**

The valueOf() method returns the primitive value of a string.

The valueOf() method does not change the original string.

The valueOf() method can be used to convert a string object into a string.

## **Syntax**

*string*.valueOf()

### **Example**

let text = "Hello World!";  
let result = text.valueOf();