String Methods in JavaScript with Examples

1. concat():

This method is used to combine two or more strings and return a new string. The original strings remain unchanged.

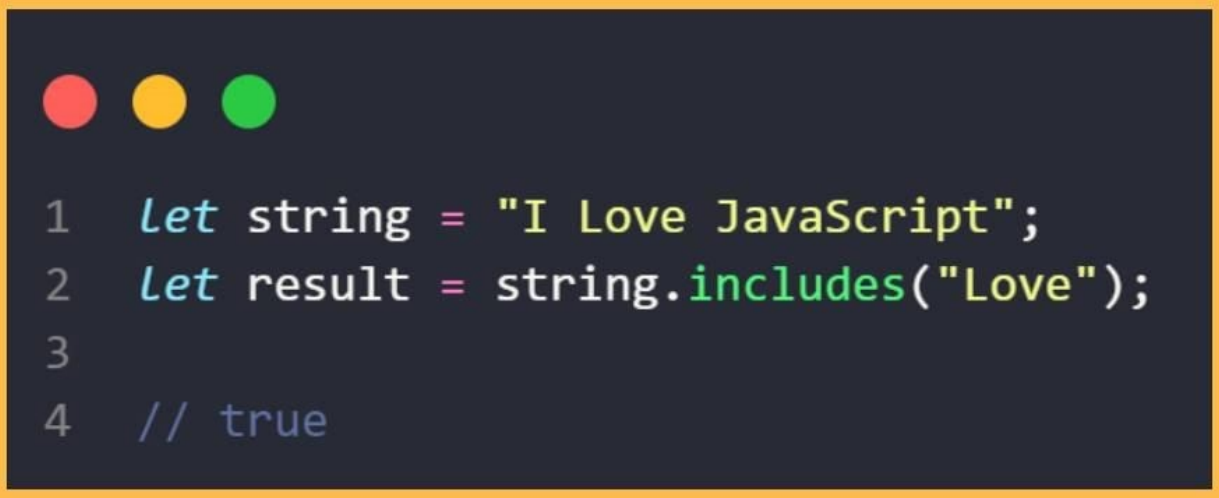
```
let string1 = "I Love ";
let string2 = "JavaScript";
let result = string1.concat(string2);

// "I Love JavaScript"
```

In our example, we're combining "I Love" and "JavaScript" to get "I Love JavaScript".

2. Includes():

This method checks whether a string contains a specified substring and returns a boolean value, true if the string contains the substring and false if it doesn't.



```
let string = "I Love JavaScript";
let result = string.includes("Love");

// true
```

In the example, we're checking if our string contains the word "Love", and the method returns true because "Love" is indeed part of our string.

3. split():

This method divides a string into an ordered list of substrings, puts these substrings into an array, and returns the array.

The division is done by searching for a specified pattern (the separator).

Here we're splitting the string into words using a space ("") as the separator.

```
let string = "I Love JavaScript";
let result = string.split(" ");

// ["I", "Love", "JavaScript"]
```

4. replace():

This method returns a new string with a replacement for some or all of the pattern matches.

The pattern can be a string or a RegExp, and the replacement can be a string or a function to be called for each match.

If the pattern is a string, only the first match will be replaced. Here we're replacing the word "Love" with "Adore" in our string.

5. replaceAll():

This method returns a new string with a replacement for each match of a pattern.

The pattern can be a string or a RegExp, and the replacement can be a string or a function to be called for each match.

```
let string = "I Love JavaScript, I Really Do!";
let result = string.replaceAll("I", "We");

// "We Love JavaScript, We Really Do!"
```

In this example, we replace all instances of "I" with "We" in our string.

6. trim():

This method removes whitespace from both ends of a string. Whitespace in this context is all the whitespace characters (space, tab, no-break space, etc.) and all the line terminator characters.

We use this on a string with extra spaces at the start and end, and it returns the string without those.

```
let string = " I Love JavaScript ";
let result = string.trim();

// "I Love JavaScript"
```

7. toUpperCase():

This method returns the calling string value converted to uppercase.

```
let string = "I Love JavaScript";
let result = string.toUpperCase();

// "I LOVE JAVASCRIPT"
```

In the example, our string "I Love JavaScript" becomes "I LOVE JAVASCRIPT".

8. toLowerCase():

This method returns the calling string value converted to lowercase. So, "I Love JavaScript" turns into "i love javascript".