MDN Web Docs
moz://a

String.prototype.replaceAll()

The `replaceAll()` method returns a new string with all matches of a `pattern` replaced by a `replacement`. 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.

The original string is left unchanged.

JavaScript Demo: String.replaceAll()

```
const p = 'The quick brown fox jumps over the lazy dog. If the dog reacted, was

console.log(p.replaceAll('dog', 'monkey'));
// expected output: "The quick brown fox jumps over the lazy monkey. If the mon


// global flag required when calling replaceAll with regex
const regex = /Dog/ig;
console.log(p.replaceAll(regex, 'ferret'));
// expected output: "The quick brown fox jumps over the lazy ferret. If the fer

```

Run ›

Reset

Syntax

```
replaceAll(regexp, newSubstr)
replaceAll(regexp, replacerFunction)

replaceAll(substr, newSubstr)
replaceAll(substr, replacerFunction)
```

Note: When using a `*regexp*` you have to set the global ("g") flag; otherwise, it will throw a

> `TypeError`: "replaceAll must be called with a global RegExp".

Parameters

`regexp` (pattern)

A `RegExp` object or literal with the global flag. The matches are replaced with `newSubstr` or the value returned by the specified `replacerFunction`. A RegExp without the global ("g") flag will throw a `TypeError`: "replaceAll must be called with a global RegExp".

`substr`

A `String` that is to be replaced by `newSubstr`. It is treated as a literal string and is *not* interpreted as a regular expression.

`newSubstr` (replacement)

The `String` that replaces the substring specified by the specified `regexp` or `substr` parameter. A number of special replacement patterns are supported; see the "Specifying a string as a parameter" section below.

`replacerFunction` (replacement)

A function to be invoked to create the new substring to be used to replace the matches to the given `regexp` or `substr`. The arguments supplied to this function are described in the "Specifying a function as a parameter" section below.

Return value

A new string, with all matches of a pattern replaced by a replacement.

Description

This method does not change the calling `String` object. It returns a new string.

Specifying a string as a parameter

The replacement string can include the following special replacement patterns:

Pattern	Inserts
`$$`	Inserts a `"$"`.
`$&`	Inserts the matched substring.
`$``	Inserts the portion of the string that precedes the matched substring.

Pattern	Inserts
`$'`	Inserts the portion of the string that follows the matched substring.

`$n`	Where `n` is a positive integer less than 100, inserts the `n` th parenthesized submatch string, provided the first argument was a `RegExp` object. Note that this is `1` -indexed.

Specifying a function as a parameter

You can specify a function as the second parameter. In this case, the function will be invoked after the match has been performed. The function's result (return value) will be used as the replacement string. (**Note:** The above-mentioned special replacement patterns do *not* apply in this case.)

Note that if the first argument of an `replaceAll()` invocation is a `RegExp` object or regular expression literal, the function will be invoked multiple times.

The arguments to the function are as follows:

Possible name	Supplied value
`match`	The matched substring. (Corresponds to `$&` above.)
`p1, p2, ...`	The *n*th string found by a parenthesized capture group, provided the first argument to `replaceAll()` was a `RegExp` object. (Corresponds to `$1` , `$2` , etc. above.) For example, if `/(\a+)(\b+)/` , was given, `p1` is the match for `\a+` , and `p2` for `\b+` .
`offset`	The offset of the matched substring within the whole string being examined. (For example, if the whole string was `'abcd'` , and the matched substring was `'bc'` , then this argument will be `1` .)
`string`	The whole string being examined.
`namedGroups`	An object of all named capturing groups. The keys are the names of the capturing groups and each value is the substring matching the named capture group. If the regular expression doesn't contain any capturing groups, `namedGroups` is undefined.

(The exact number of arguments depends on whether the first argument is a `RegExp` object—and, if so, how many parenthesized submatches it specifies.)

Examples

Using replaceAll

```
'aabbcc'.replaceAll('b', '.');
// 'aa..cc'
```

Non-global regex throws

When using a regular expression search value, it must be global. This won't work:

```
'aabbcc'.replaceAll(/b/, '.');
TypeError: replaceAll must be called with a global RegExp
```

This will work:

```
'aabbcc'.replaceAll(/b/g, '.');
"aa..cc"
```

Specifications

Specification
ECMAScript Language Specification (ECMAScript) # sec-string.prototype.replaceall

Browser compatibility

Report problems with this compatibility data on GitHub

replaceAll	
Chrome	85

Edge	85
Firefox	77
Internet Explorer	**No**
Opera	71
Safari	13.1
WebView Android	85
Chrome Android	85
Firefox for Android	79
Opera Android	60
Safari on iOS	13.4
Samsung Internet	14.0
Deno	1.2
Node.js	15.0.0

 Full support

 No support

See also

- A polyfill of `String.prototype.replaceAll` is available in `core-js`
- `String.prototype.replace()`
- `String.prototype.match()`
- `RegExp.prototype.exec()`
- `RegExp.prototype.test()`

Last modified: Dec 12, 2021, by MDN contributors