String.prototype.search()

Summary: in this tutorial, you'll learn how to use the JavaScript String `search()` function to search for a match of a string with a regular expression and return the first match.

Introduction to the JavaScript String search() function

The `search()` method matches a string with a regular expression and returns the index of the first match in the string:

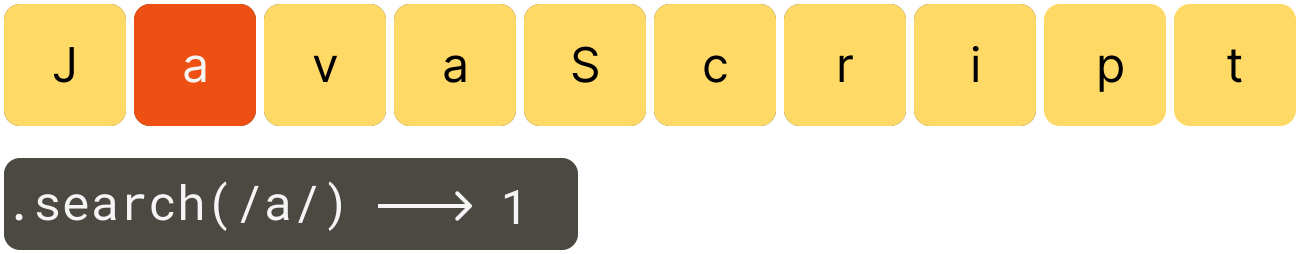
```
let index = str.search(regexp);
```

In this syntax, the `regexp` is a regular expression object.

If you pass a value that is not a `RegExp` object to the method, it will convert that value to a `RegExp` object using the following expression:

```
RegExp(regexp)
```

If the `search()` doesn't find any match, it returns -1.



Besides a regular expression object, you can pass any object that has `Symbol.search()` method to the `search()` method.

JavaScript String search() method examples

Let's take some examples of using the JavaScript `search()` function.

Basic JavaScript string search() method usages

The following example uses the `search()` method to return the first occurrence of a capital letter in a string:

```
let re = /[A-Z]/;
let str = 'hi There! How are you?';
let index = str.search(re);

console.log(index);
```

Output:

```
3
```

It returns `3` which is the index of the capital letter `T` .

The following example returns `-1` because there is no number in the string:

```
let re = /[0-9]/;
let str = 'Hello, JavaScript!';
let index = str.search(re);

console.log(index);
```

Output:

```
-1
```

Passing an object with the Symbol.search method

The following example illustrates how to use the `search()` method with an object that has the `Symbol.search` method:

```
class EmailDomainChecker {
  constructor(domain) {
    this.domain = domain;
  }

  [Symbol.search](email) {
    if (!email.includes("@")) {
      return -1;
    }
    const [, domain] = email.split("@");
    return domain === this.domain ? 1 : -1;
  }
}

const js = new EmailDomainChecker("javascripttutorial.net");

let isJsEmail = "hi@javascripttutorial.net".search(js) === 1;
console.log({ isJsEmail });

isJsEmail = "test@test.com".search(js) === 1;
console.log({ isJsEmail });
```

Output:

```
{ isJsEmail: true }
{ isJsEmail: false }
```

How it works.

First, define a class `EmailDomainChecker` with a `Symbol.search` method that returns 1 if the email has a domain that matches with a specified domain, or -1 otherwise.

```
class EmailDomainChecker {
  constructor(domain) {
    this.domain = domain;
  }
```

```
  [Symbol.search](email) {
    if (!email.includes("@")) {
      return -1;
    }
    const [, domain] = email.split("@");
    return domain === this.domain ? 1 : -1;
  }
}
```

Second, create a new EmailDomainChecker object with the `gmail.com` domain:

```
const js = new EmailDomainChecker("javascripttutorial.net");
```

Third, check if the email `hi@javascriptutorial.net` comes from the `javascripttutorial.net` domain using the `search()` method:

```
let isJsEmail = "hi@javascripttutorial.net".search(js) === 1;
console.log({ isJsEmail });
```

The search() method will invoke the `Symbol.search` method of the `js` object that returns 1 in this case.

Finally, check if the `test@test.com` email belongs to the domain name `javascripttutorial.net` :

```
isJsEmail = "test@test.com".search(js) === 1;
console.log({ isJsEmail });
```

It returns -1 in this case.

Summary

- Use the JavaScript String `search()` to find the index of the first match based on a regular expression in a string.