MDN web docs
moz://a
Connexion
Rechercher sur MDN
Technologies ▾
Guides et références ▾
Votre avis ▾

String.prototype.match()

Français ▾

La méthode **`match()`** permet d'obtenir le tableau des correspondances entre la chaîne courante et une expression rationnelle.

JavaScript Demo: String.match()

```
1 var paragraph = 'The quick brown fox jumps over the lazy dog. It barked.';
2 var regex = /[A-Z]/g;
3 var found = paragraph.match(regex);
4
5 console.log(found);
6 // expected output: Array ["T", "I"]
```

Run ›

Reset

Syntaxe

```
str.match(regexp)
```

Paramètres

`regexp`
Un objet représentant une expression rationnelle. Si ce n'est pas un objet de type `RegExp`, celui-ci sera converti en un objet `RegExp` grâce à `new RegExp(regexp)`. Si aucun paramètre n'est utilisé, cela renverra un tableau contenant un élément étant la chaîne vide : `[""]`.

Valeur de retour

Un tableau (`Array`) contenant les correspondances et les groupes capturés avec les parenthèses ou `null` s'il n'y a pas de correspondance. Le contenu de ce tableau dépend de l'utilisation du marqueur pour la recherche globale `g` :

- Si le marqueur `g` est utilisé, tous les résultats correspondants à l'expression rationnelle complète seront renvoyés mais les groupes capturants ne seront pas renvoyés.
- Si le marqueur `g` n'est pas utilisé, seule la première correspondance et ses groupes capturants seront renvoyés. Dans ce cas, l'élément renvoyé aura des propriétés supplémentaires listées ci-après.

Propriétés supplémentaires

Comme indiqué ci-avant, les résultats peuvent contenir certaines propriétés supplémentaires :

- `groups` : un tableau de groupes capturants nommés ou `undefined` si aucun groupe capturant n'a été défini. Voir la page sur les groupes et les intervalles pour plus d'informations.
- `index` : l'indice de la chaîne de caractères où a été trouvée la correspondance.
- `input` : une copie de la chaîne sur laquelle a été effectuée la recherche.

Description

Si l'expression n'utilise pas le drapeau (*flag*) `g`, le résultat obtenu sera le même qu'avec `RegExp.exec()`.

Voir aussi : les méthodes de `RegExp`

- Si on souhaite savoir s'il existe des correspondances entre une chaîne de caractères et une expression rationnelle `RegExp`, on pourra utiliser `RegExp.test()`.
- Si on ne souhaite obtenir que la première correspondance, on pourra plutôt utiliser `RegExp.exec()` à la place.
- Si on souhaite obtenir les groupes correspondants et que le drapeau « global » est activé, il faudra utiliser `RegExp.exec()` à la place.

Exemples

Utiliser `match()`

Dans l'exemple suivant, on utilise `match()` afin de trouver la chaîne `'Chapitre'` suivie par un ou plusieurs chiffres séparés par des points. L'expression utilisée active le drapeau `i` afin que la casse ne soit pas prise en compte.

```
var str = 'Pour plus d\'informations, voir le chapitre 3.4.5.1';
var re = /(chapitre \d+(\.\d)*)/i;
var trouvé = str.match(re);

console.log(trouvé);

// logs ['chapitre 3.4.5.1', 'chapitre 3.4.5.1', '.1']

// 'chapitre 3.4.5.1' est la première correspondance
// 'chapitre 3.4.5.1' est la valeur gardée en mémoire par
// `(chapitre \d+(\.\d)*)`.
// '.1' est la valeur gardée en mémoire par `(\.\d)`.
```

Utiliser les drapeaux `g` (global) et `i` (ignorer la casse) avec `match()`

Dans cet exemple, on illustre comment utiliser des drapeaux avec l'expression rationnelle qui est un argument de `match()`. Chaque lettre de A à E et de a à e est renvoyée, chacune dans un élément du tableau de résultat.

```
var str = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz';
var regexp = /[A-E]/gi;
var tableau_correspondances = str.match(regexp);

console.log(tableau_correspondances);
// ['A', 'B', 'C', 'D', 'E', 'a', 'b', 'c', 'd', 'e']
```

Utiliser un paramètre qui n'est pas une RegExp

Lorsque le paramètre passé à la fonction est une chaîne de caractères ou un nombre, il est converti de façon implicite en un objet `RegExp` grâce à `new RegExp(obj)`. Si c'est un nombre positif avec le signe +, la méthode `RegExp()` ignorera ce signe.

```
var str1 = "NaN signifie : qui n'est pas un nombre.";
var str2 = "Mon père a 65 ans."
str1.match("nombre");   // "nombre" est une chaîne, renvoie ["nombre"]
str1.match(NaN);        // NaN est de type number, renvoie ["NaN"]
str2.match(65);         // Renvoie ["65"]
str2.match(+65);        // Renvoie également ["65"]
```

Spécifications

Spécification	État	Commentaires
ECMAScript 3rd Edition (ECMA-262)	ST Standard	Définition initiale. Implémentée avec JavaScript 1.2.
ECMAScript 5.1 (ECMA-262) La définition de 'String.prototype.match' dans cette spécification.	ST Standard	
ECMAScript 2015 (6th Edition, ECMA-262)	ST Standard	

La définition de 'String.prototype.match' dans cette spécification.		
ECMAScript Latest Draft (ECMA-262) La définition de 'String.prototype.match' dans cette spécification.	D Projet	

Compatibilité des navigateurs

Update compatibility data on GitHub

match	
Chrome	Oui
Edge	12
Firefox	1
IE	Oui
Opera	Oui
Safari	Oui
WebView Android	Oui
Chrome Android	Oui
Firefox Android	4
Opera Android	Oui
Safari iOS	Oui
Samsung Internet Android	Oui
nodejs	Oui

flags	
Chrome	Non
Edge	Non
Firefox	1 — 49
IE	Non

Opera	Non
Safari	Non
WebView Android	Non
Chrome Android	Non
Firefox Android	4 — 49
Opera Android	Non
Safari iOS	Non
Samsung Internet Android	Non
nodejs	Non

What are we missing?

.. Support complet

.. Aucun support

Fonctionnalité non-standard. Celle-ci peut être incorrectement supportée par les autres navigateurs.

Obsolète. Les nouveaux sites web ne doivent pas utiliser cette fonctionnalité.

Notes spécifiques à Firefox/Gecko

- `flags` était un second argument non standard présent uniquement sur Gecko : *str*.match(*regexp, flags*) et a été retiré avec Firefox 49.
- À partir de Firefox 27, cette méthode a été ajustée afin d'être conforme à ECMAScript. Lorsque `match()` est appelée sur une expression rationnelle globale, la propriété `RegExp.lastIndex` de l'objet sera redéfini à `0` (bug 501739).

Voir aussi

- `RegExp`
- `RegExp.prototype.exec()`
- `RegExp.prototype.test()`

Dernière modification : 23 avr. 2019, by MDN contributors

Sujets associés

Objets standards

`String`

Propriétés

`String.prototype`

`String.length`

Méthodes

`String.fromCharCode()`

`String.fromCodePoint()`

`String.prototype.anchor()`

`String.prototype.big()`

`String.prototype.blink()`

`String.prototype.bold()`

`String.prototype.charAt()`

`String.prototype.charCodeAt()`

`String.prototype.codePointAt()`

String.prototype.concat()

String.prototype.endsWith()

String.prototype.fixed()

String.prototype.fontcolor()

String.prototype.fontsize()

String.prototype.includes()

String.prototype.indexOf()

String.prototype.italics()

String.prototype.lastIndexOf()

String.prototype.link()

String.prototype.localeCompare()

String.prototype.match()

String.prototype.matchAll()

String.prototype.normalize()

String.prototype.padEnd()

String.prototype.padStart()

String.prototype.quote()

String.prototype.repeat()

String.prototype.replace()

String.prototype.search()

String.prototype.slice()

String.prototype.small()

String.prototype.split()

String.prototype.startsWith()

String.prototype.strike()

String.prototype.sub()

String.prototype.substr()

String.prototype.substring()

String.prototype.sup()

String.prototype.toLocaleLowerCase()

String.prototype.toLocaleUpperCase()

String.prototype.toLowerCase()

String.prototype.toSource()

String.prototype.toString()

String.prototype.toUpperCase()

String.prototype.trim()

String.prototype.trimEnd()

String.prototype.trimStart()

String.prototype.valueOf()

String.prototype[@@iterator]()

String.raw()

Héritage :

`Function`

Propriétés

Function.arguments

Function.arity

Function.caller

Function.displayName

Function.length

Function.name

Function.prototype

Méthodes

Function.prototype.apply()

Function.prototype.bind()

Function.prototype.call()

Function.prototype.isGenerator()

Function.prototype.toSource()

Function.prototype.toString()

`Object`

Propriétés

Object.prototype.__count__

Object.prototype.__noSuchMethod__

Object.prototype.__parent

Object.prototype.__proto__

Object.prototype.constructor

Méthodes

Object.prototype.__defineGetter__()

Object.prototype.__defineSetter__()

Object.prototype.__lookupGetter__()

Object.prototype.__lookupSetter__()

Object.prototype.hasOwnProperty()

Object.prototype.isPrototypeOf()

Object.prototype.propertyIsEnumerable()

Object.prototype.toLocaleString()

Object.prototype.toSource()

Object.prototype.toString()

Object.prototype.unwatch()

Object.prototype.valueOf()

Object.prototype.watch()

Object.setPrototypeOf()

Recevez le meilleur du développement web

Get the latest and greatest from MDN delivered straight to your inbox.

Cette lettre d'information est uniquement disponible en anglais pour l'instant.

vous@example.com

Sign up now