

MDN web docs

[Technologies](#) ▼[Guides et références](#) ▼[Votre avis](#) ▼[Connexion](#) 

JSON

L'objet **JSON** contient des méthodes pour interpréter du [JSON](#) (JavaScript Object Notation) (voir également la page du glossaire [JSON](#)) et convertir des valeurs en JSON. Il ne peut être appelé ou construit, et, en dehors de ses deux méthodes, n'a pas de fonctionnalité propre.

Différences entres JavaScript et JSON

JSON est une syntaxe pour sérialiser des objets, tableaux, nombres, chaînes de caractères, booléens et valeurs *null*. Elle est basée sur la syntaxe de JavaScript mais en est distincte : du code JavaScript n'est pas nécessairement du JSON, et du JSON n'est pas nécessairement du JavaScript.

- Pour les objets et les tableaux
 - Les noms de propriété doivent être des chaînes de caractères délimitées par des guillemets doubles ; les *trailing commas* sont interdits
- Pour les nombres
 - Les zéros non significatifs sont interdits ; un point décimal doit être suivi d'au moins un chiffre (plus exactement : `JSON.stringify()` ignorera les zéros mais `JSON.parse()` déclenchera une exception `SyntaxError`).
- Pour le texte : **tout texte JSON est une expression JavaScript** (pour les moteurs qui implémentent [cette proposition](#)).

- Pour les autres moteurs, seul un jeu limité de caractères peut être échappé ; certains caractères de contrôle sont interdits ; le séparateur de ligne Unicode (↵ U+2028) et le séparateur de paragraphe (↵ U+2029) sont autorisés en JSON mais pas en JavaScript dans les littéraux de chaînes de caractères.

Dans l'exemple suivant, on utilise `JSON.parse()` afin d'analyser la chaîne JSON et `eval` afin d'exécuter le code correspondant :

```
1 | var code = '"\u2028\u2029"';  
2 | JSON.parse(code); // vaut "\u2028\u2029" pour tous les moteurs  
3 | eval(code); // provoque une SyntaxError pour les anciens moteurs
```

Syntaxe complète [↗](#)

```
1 | JSON = null  
2 |     ou true ou false  
3 |     ou NombreJSON  
4 |     ou ChaîneJSON  
5 |     ou ObjetJSON  
6 |     ou TableauJSON  
7 |  
8 | NombreJSON = - NombrePositif  
9 |             ou NombrePositif  
10 | NombrePositif = NombreDécimal  
11 |                ou NombreDécimal . Chiffres  
12 |                ou NombreDécimal . Chiffres PartiExposant  
13 |                ou NombreDécimal PartiExposant  
14 | NombreDécimal = 0  
15 |                ou UnÀNeuf Chiffres  
16 | PartiExposant = e Exposant  
17 |                ou E Exposant  
18 | Exposant = Chiffres  
19 |            ou + Chiffres  
20 |            ou - Chiffres  
21 | Chiffres = Chiffre  
22 |           ou Chiffres Chiffre  
23 | Chiffre = 0 à 9
```

```

24  UnÀNeuf = 1 à 9
25
26  ChaîneJSON = ""
27      ou " ChaîneCaractères "
28  ChaîneCaractères = ChaîneCaractère
29      ou ChaîneCaractères ChaîneCaractère
30  ChaîneCaractère = un caractère
31      sauf " ou \ ou U+0000 à U+001F
32      ou SéquenceÉchappement
33  SéquenceÉchappement = \" ou \/ ou \\ ou \b ou \f ou \n ou \r ou \t
34      ou \u ChifreHexadécimal ChifreHexadécimal ChifreHexadéc
35  ChifreHexadécimal = 0 à 9
36      ou A à F
37      ou a à f
38
39  ObjetJSON = { }
40      ou { Membres }
41  Membres = ChaîneJSON : JSON
42      ou Membres , ChaîneJSON : JSON
43
44  TableauJSON = [ ]
45      ou [ ÉlémentsTableau ]
46  ÉlémentsTableau = JSON
47      ou ÉlémentsTableau , JSON

```

Des espaces blancs insignifiants peuvent être présents n'importe où sauf dans un *JSONNumber* (les nombres ne doivent pas contenir d'espaces blancs) ou dans un *JSONString* (where it is interpreted as the corresponding character in the string, or would cause an error). Les caractères tabulation (↗ [U+0009](#)), retour chariot (↗ [U+000D](#)), saut de ligne (↗ [U+000A](#)), and espace (↗ [U+0020](#)) sont les seuls caractères blancs valides.

Méthodes




JSON.parse()

Interprète une chaîne de caractères comme du JSON, transformant de façon optionnelle la valeur produite et ses propriétés, puis retourne la valeur.

JSON.stringify()

Retourne une chaîne de caractères JSON correspondant à la valeur spécifiée, incluant de façon optionnelle seulement certaines propriétés, ou remplaçant des valeurs de propriété dans une forme définie par l'utilisateur.

Spécifications

Spécification	État	Commentaires
ECMAScript 5.1 (ECMA-262) La définition de 'JSON' dans cette spécification.	 ST Standard	Définition initiale.
ECMAScript 2015 (6th Edition, ECMA-262) La définition de 'JSON' dans cette spécification.	 ST Standard	
ECMAScript Latest Draft (ECMA-262) La définition de 'JSON' dans cette spécification.	 D Projet	

Compatibilité des navigateurs

[Take this quick survey to help us improve our browser compatibility tables](#)

Support simple

Chrome	Oui
Edge	Oui
Firefox	3.5
IE	8
Opera	10.5
Safari	4
WebView Android	Oui
Chrome Android	Oui
Edge Mobile	Oui